Calhoun
Institutional Archive of the Naval Postgraduate School

Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations 1. Thesis and Dissertation Collection, all items

1982-06

Evaluation of a highly skewed propeller for a Naval auxiliary (AO-177).

Jamison, Richard L.

Massachusetts Institute of Technology

http://hdl.handle.net/10945/20154

Downloaded from NPS Archive: Calhoun

DUDLEY KNOX LIBRARY

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

http://www.nps.edu/library

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIF. 93940

EVALUATION OF A HIGHLY SKEWED PROPELLER FOR A NAVAL AUXILIARY (AO-177)

by

RICHARD L. JAMISON
LIEUTENANT, U.S. NAVY
S.B., Massachusetts Institute of Technology
(1976)

SUBMITTED TO THE DEPARTMENT OF OCEAN
ENGINEERING IN PARTIAL FULFILLMENT
OF THE DEGREE OF OCEAN ENGINEER
AND THE THE DEGREE OF MASTER OF
SCIENCE IN NAVAL ARCHITECTURE
AND MARINE ENGINEERING
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June, 1982

© Massachusetts Institute of Technology 1982

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIF. 93940

EVALUATION OF A HIGHLY SKEWED PROPELLER FOR A NAVAL AUXILIARY (AO-177)

by

Richard L. Jamison

Submitted to the Department of Ocean Engineering on May 7, 1982, in partial fulfillment of the requirements for the degree of Ocean Engineer and the degree of Master of Science in Naval Architecture and Marine Engineering.

ABSTRACT

The cavitation performance of a seven-bladed, highly skewed propeller for a Naval auxiliary (AO-177) is evaluated using lifting surface numerical hydrodynamic methods. An important contribution to accurate predition of cavitation performance is shown to be an accurate model of the effective wake.

A new model of the effective wake is described. A program to calculate unsteady time-averaged, but not circumferentially averaged, field point velocities is presented. The propeller-induced velocities, along with the original nominal wake, are combined with Huang's axisymmetric effective wake scheme in pie-shaped wake segments to determine the effective wake.

The method is used to predict severe cavitation extent. This prediction is confirmed by SSPA experiments.

Thesis Supervisor: David V. Burke
Title: Professor of Naval Architecture

ACKNOWLEDGEMENT

The author wishes to express his gratitude to the Navy for supporting this thesis. In particular, M. Wilson and R. Boswell of DTNSRDC and A. Chase of NAVSEA provided the needed raw data and background information in a timely manner and expressed strong support by way of their interest.

Thanks are extended to Professor D. V. Burke, who served as thesis advisor and who also provided the original motivation for the project, and to Professors J. E. Kerwin and R. Van Houten, whose experience and physical insight invariably got the project moving again each time it bogged down.

I would also like to express my appreciation to my wife, Beverly, not only for her support of our home, but also for her technical assistance in teaching me programming techniques and debugging tricks.

Finally, I wish to thank my daughter, Ruth, just for being Ruth.

TABLE OF CONTENTS

TITLE PAGE

Abstract 2

Acknowledgement 3

Table of Contents 4

I. Introduction 5

II. Impact of Effective Wake 7

III. Method of Effective Wake Calculation 10

IV. AO-177 Effective Wake Calculations 15

V. Conclusions 19

VI. Recommendations 20

Appendices

I. Nominal, Induced, and Effective Velocities 21

II. Nominal, Induced, and Effective Velocities, Non-Dimensionalized 28

III. Propeller Cavitation, Nominal Wake 35

IV. Propeller Cavitation, Effective Wake 44

V. Cavity Length, Nominal and Effective Wakes 53

VI. Cavity Volume, Nominal and Effective Wakes 55

VII. SSPA Experimental Propeller Cavitation Extent Plots 57

VIII. Nominal Wake Diagram 60

IX. Effective Wake Diagram 62

X. UFPV Program Listing 64

XI. PIEWAKE Program Listing 89

References 98

I. INTRODUCTION

The David W. Taylor Naval Ship Research and Development Center has recently designed a seven-bladed, highly skewed propeller for a Naval Auxiliary (AO-177). This design is instructive concerning the state of the art in computer-aided propeller design.

The history of the propeller design is chronicled in reports by Boswell (1), Valentine and Chase (2), and Hendrican and Remmers (3). Computer-aided propeller applications included lifting line theory, used to determine the radial load distribution and the radial hydrodynamic pitch angle, and lifting surface theory, used to determine the final geometric pitch distribution, camber distribution, and final propeller offsets, including fillets, trailing and leading edge details, additional thickness added to the trailing edge, tip geometry, rake, and hub details. Cavitation performance was predicted by the method of Burrill and Emerson (4) and was also evaluated by experiment.

However, sea trials of the AO-177 indicated a severe unsteady propeller force problem. Inspection of the propeller indicated that it was cavitating significantly worse than had been predicted. The Burrill and Emerson method, originally designed for

four-to-six-bladed propellers with an expanded area ratio of 0.60, did not extend well to the seven-bladed AO-177 propeller with an expanded area ratio of 0.77, even though both were for merchant-type hulls. Further, scale effects could have thrown off the model tests.

II. IMPACT OF EFFECTIVE WAKE

The AO-177 propeller performance was evaluated at MIT using some numerical propeller analysis computer programs developed at MIT. Kerwin and Lee (5) describe MIT-PUF-2, a computer program to calculate forces, steady and unsteady, generated by a propeller. MIT-FPV, a steady field point velocity program, is described in a report by Min (6). Lee (7) reports on MIT-PUF-3, a computer program to predict steady and unsteady propeller cavitation extent and volume. All three programs model the propeller as a grid of discrete vortex segments. The wake is also modelled as a vortex grid in each program.

Cavitation performance calculations, more than propeller force and induced velocity calculations, are highly sensitive to the wake data used as input. Experience at MIT and elsewhere has shown that using the nominal wake data to approximate inflow conditions, while acceptable for propeller force and induced velocity calculations, is not acceptable for cavitation performance calculations. The wake velocities measured behind a model in the absence of any propeller (the nominal wake) must be modified to account for change in boundary conditions imposed by the presence of a propeller. This new, modified wake is generally called the "effective wake." There is, unfortunately, no

general method for anticipating the effective wake generated by a particular propeller in a general nominal wake.

Huang and others (8,9) have developed both theory and numerical schemes for calculating the effective wake of a propeller in an axisymmetric nominal wake. Det Norske Veritas has attempted to adapt Huang's method to non-axisymmetric wakes, such as surface ship wakes, by dividing the nominal wake into pie-shaped segments and then applying the axisymmetric wake calculation within each pie segment, using the nominal wake and the steady propeller induced velocities. This approach is an improvement over blindly applying the axisymmetric effective wake calculation to a surface ship's wake, but, owing to the very sharp AO-177 wake (even by surface ship standards), it was judged that some further refinement of Huang's method was required.

Huang's method assumed an axisymmetric nominal wake and a steady propeller-induced velocity field. The Det Norske Veritas method assumed a general nominal wake and a steady propeller-induced velocity field. Short of a completely general effective wake calculation, the most obvious refinement was to assume a general nominal wake, but allow an unsteady propeller-induced velocity field. This was the method

used for this paper.

The current method of effective wake calculations can be summarized in the following steps:

1) Use the nominal wake and propeller data to calculate the unsteady propeller-induced velocity field at representative points both inside and outside the propeller disk.

2) Divide the nominal wake into small pie-shaped segments as in the Det Norske Veritas method.

3) Calculate the effective wake velocities within each pie segment using the nominal wake velocities and the unsteady propeller-induced velocities, assuming that each pie segment acts as though it were part of an axisymmetric wake with the same radial velocity distribution.

III. METHOD OF EFFECTIVE WAKE CALCULATION

The two new aspects of the current effective wake calculations are the propeller-induced unsteady field point velocity (UFPV) calculation and the use of these time-averaged but not circumferentially-averaged velocities in a pie segment nominal wake modification calculation (PIEWAKE). Each of these two new methods will be illustrated by a typical calculation.

The unsteady field point velocity (UFPV) program is basically a generalization of the steady field point velocity program (MIT-FPV) by Min. (6). The unsteady version uses the full wake lattice arrangement also employed in MIT-PUF-2. Program inputs are the FILE14 DATA file output from MIT-PUF-2 and interactive directions. Program outputs are either single point calculations directed to the user's terminal or a data file to be used as input into PIEWAKE, for effective wake calculations.

UFPV first reads the FILE14 DATA input, containing mostly propeller geometry, singularity strengths, and singularity geometry. It should be noted that the number of propeller revolution time steps specified in MIT-PUF-2 must be divisible by the number of propeller blades . For effective wake calculations, the wake input to MIT-PUF-2 ought strictly to be the effective wake; however, using the nominal wake to calculate the propeller-induced velocities

is a good approximation.

The user next inputs field point coordinates. The field point position may be entered in cylindrical coordinates or in terms of the rotating propeller and wake grids.

The velocity induced by one blade at each time step is then calculated. At each time step, the vortex lattice is assigned the strengths corresponding to the time step of the propeller rotation. The velocity induced by each element of the vortex grid, both blade and wake, is then summed to obtain the total velocity induced by one blade in that particular angular position.

The velocities induced by each blade of the propeller are then summed to produce the total propeller-induced velocity for that position of the propeller. Velocities are calculated for different propeller positions and then harmonically analyzed. The zeroeth harmonic, then, is the time-averaged propeller-induced velocity for that field point. (By averaging all points at a given radius we could obtain the circumferentially-averaged, time-averaged, propeller-induced velocity, the output of the steady MIT-FPV.)

For effective wake calculations, such velocities are computed for up to 60 points per radius for radii varying from propeller hub to 1.7 times the propeller radius.

Between the propeller hub and propeller tip, the velocities are calculated at the leading edge panel on the propeller grid.

For the AO-177 propeller analysis, 56 points per radius for eight radii were used to create a file of unsteady field point velocities.

PIEWAKE is a program to perform Huang's axisymmetric effective wake contraction calculations in each of many pie-shaped segments of the wake. It uses nominal wake data and unsteady propeller-induced velocity data from UFPV as inputs to produce an effective wake velocity data file as output.

After reading nominal wake data and the induced velocity field data, PIEWAKE extrapolates the nominal wake data to the hub. The innermost radius of the induced velocity field is taken as the hub. (UFPV had taken, as its innermost radius, the hub radius used in MIT-PUF-2 and passed to UFPV by way of FILE14 data file.)

In each pie segment, Huang's method is used to calculate an effective velocity and the effective radius corresponding to that effective velocity. In the numerical form, this method is iterative in nature.

A finite difference equation, given in reference (9), is used to calculate the effective velocity. Given the

nominal wake at radii (assumed, for the first iteration, to be the nominal radii) in the presence of a propeller, an effective velocity can be calculated if an effective velocity at the next radius away from the hub is known. The effective velocity at the outermost radius is assumed to be equal to the nominal velocity there, enabling all effective velocities to be calculated.

A second finite difference equation, also given in reference (9), is used to calculate the effective radius corresponding to the set of effective velocities just calculated. This equation uses nominal radii, nominal velocities, and apparent velocities. (An apparent velocity is the sum of the effective velocity and the propeller-induced velocity at a point.) Given one effective radius, the next radius away from the hub can be calculated. Since the innermost radius of all velocity fields is set at the hub radius, all effective radii can be determined sequentially from innermost to outermost. These radii can then be used in the finite difference equation to calculate the next iteration of effective velocities. PIEWAKE iterates five times for each pie segment.

In addition to the effective velocity wake field, PIEWAKE also computes the mean velocity, the volumetric mean velocity of the nominal wake, the volumetric mean velocity of the effective wake, and the effective blockage.

The effective velocity field then then be processed

and used by MIT-PUF-3 as any other wake field.

IV. AO-177 EFFECTIVE WAKE CALCULATIONS

Appendix I contains plots of nominal, induced, and effective velocities varying circumferentially at several specified radii. Appendix II contains the same information, except that the nominal and effective velocities are normalized on the respective volumetric mean velocity. These plots confirm that the modifications to the nominal wake outlined in Chapter III are indeed reasonable. Several salient conclusions can be drawn from these plots:

1) Effective velocity modifications of the nominal velocities are in the expected direction. Physically, the imposition of the propeller on the nominal wake draws more water into the slipstream, thus increasing the velocities. This "faster water" will come from radii greater than the field point being sampled; in other words, the wake will seem to "contract."

2) Effective velocity modifications of the nominal wake velocities are greatest at lower radii. The wake "contraction" will seem to concentrate here.

3) In general, the wake peak is more narrow in the effective wake than in the nominal wake. In the case of the AO-177 nominal wake, the existence in the presence of a propeller of a wake peak as sharp as that in the nominal wake is counter-intuitive. Therefore, a decrease in the wake peak is to be expected.

4) The normalized effective velocities at 0 degrees

are not appreciably different from that of the nominal wake. This is disappointing and was not expected. It may be a result of radial vorticity, ignored by the axisymmetric effective wake calculation performed in each pie segment. This assumption is least appropriate in the 0 degree region of the wake. A more general effective wake calculation scheme may be required to obtain better results in this region.

Results of MIT-PUF-3 cavitation performance of the AO-177 propeller are shown in Appendices III and IV. As an indicator of the influence of effective wake modifications to the nominal wake, MIT-PUF-3 was run twice, once with the nominal wake as input and once with the effective wake (the more accurate boundary condition) as input. Appendix III shows cavitation performance given the nominal wake as input while Appendix IV shows cavitation performance given the effective wake as input. In general, the plots show that the cavitation extent is not changed much, although cavitation performance is better in the effective wake at the lower radii where, as has been shown, the effective wake modifications are the greatest. At the outer radii, where the effective wake modification is the least significant, the cavitation performance is least changed between the nominal wake and effective wake cases.

Appendix V contains a graph comparing cavitation extent of MIT-PUF-3 output given nominal wake input against that given effective wake input. It plots the length of

cavitation on the seventh radial cavitation panel (about the .8 propeller radius) non-dimensionalized on the local chord length as the blade rotates through 360 degrees. This graph indicates that the effective wake modification did not reduce cavitation extent on a blade, a result noted earlier. However, a larger contrast between the nominal wake and effective wake cases is shown in Appendix VI which graphs the cavitation volume on a blade in the two cases. The volume is non-dimensionalized on the cube of the propeller radius. This graph shows a drop of approximately 35% of peak cavitation volume. This change in cavity volume demonstrates the sensitivity of cavitation performance to small changes in the wake profile. This confirms that efforts to refine effective wake calculations were indeed justified.

The effective wake cavitation performance indicated by MIT-PUF-3 compares reasonably well with experiments performed by the SSPA propeller tunnel. These experiments measured cavitation performance of the AO-177 propeller by photographing model tests. While no cavitation volume results were obtained, cavitation extent could be compared with MIT-PUF-3 predictions. Despite reductions of the effective wake calculations compared with nominal wake calculations, MIT-PUF-3 still tended to predict greater cavitation than the experiments indicated, particularly at the middle radii. This seems to indicate that the effective wake modifications, while tending to the proper direction, were not large enough. This is consistent with

the conclusions from comparing the two wakes that a more general effective wake scheme including radial vorticity in high wake regions is required.

V. CONCLUSIONS

1) Calculation of an effective wake using Huang's axisymmetric method in pie-shaped segment with unsteady propeller-induced velocities is a reasonable approximation. It still tends to underestimate the effective wake modification to the nominal wake in the vicinity of 0 degrees. To correct this, a more general effective wake calculation scheme including radial vorticity effects may be required.

2) The current effective wake calculation scheme can be used to predict cavitation performance for highly skewed propellers with up to seven blades.

3) Cavitation performance is indeed sensitive to small changes in the wake profile. Seemingly small improvements in effective wake calculations yield significantly more accurate cavitation performance predictions.

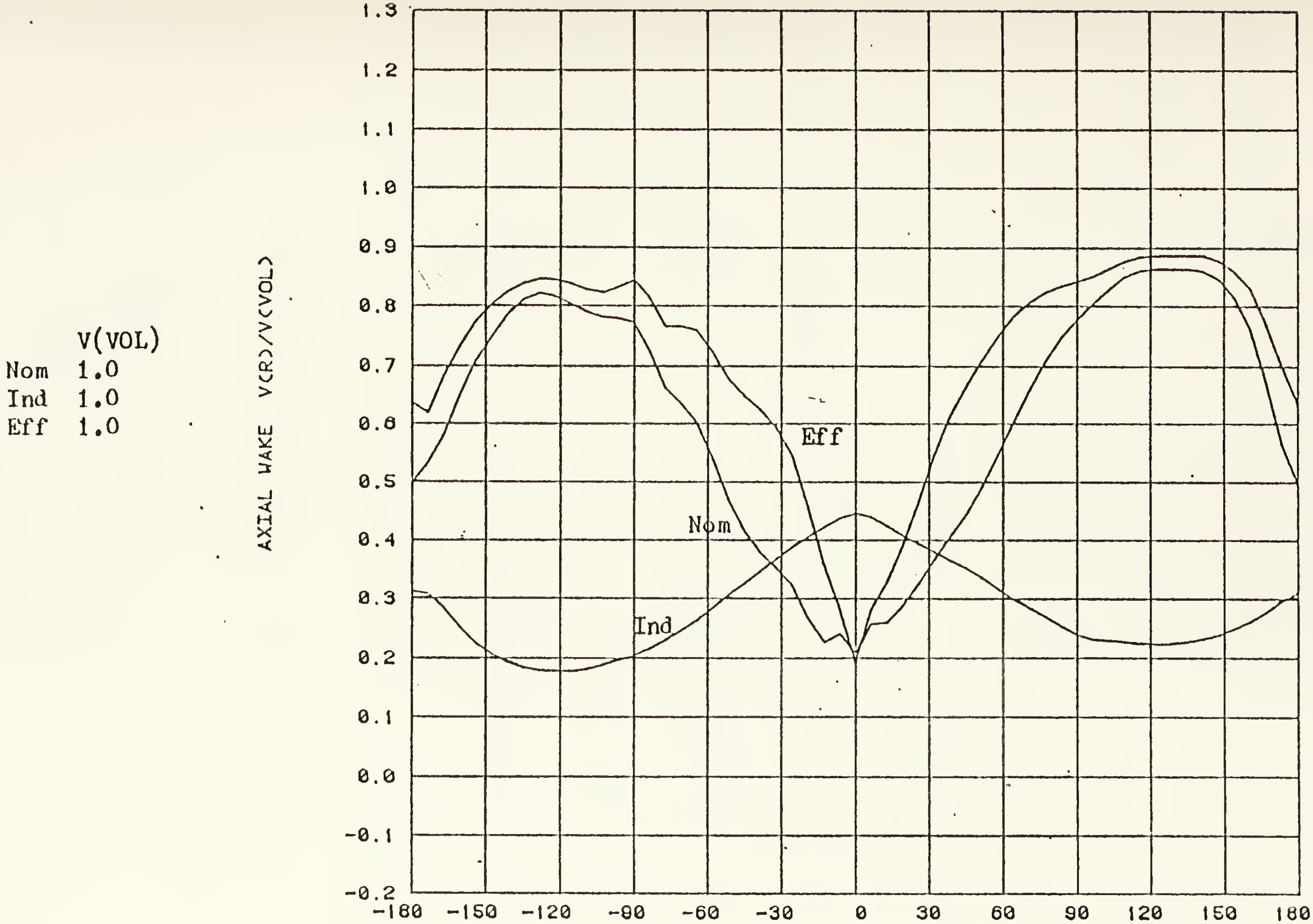
VI. RECOMMENDATIONS

1) A more general effective wake calculation scheme, including the effects of radial vorticity, is required.

2) The computational time (and hence cost) of the unsteady field point velocity program (UFPV) can probably be cut without significant sacrifice of accuracy by decreasing the number of points calculated per radius. Appendix I shows that the variation of induced velocity with rotation is slight and smooth. Calculation of fewer points and estimation of the remaining points using harmonic analysis would probably not reduce accuracy.

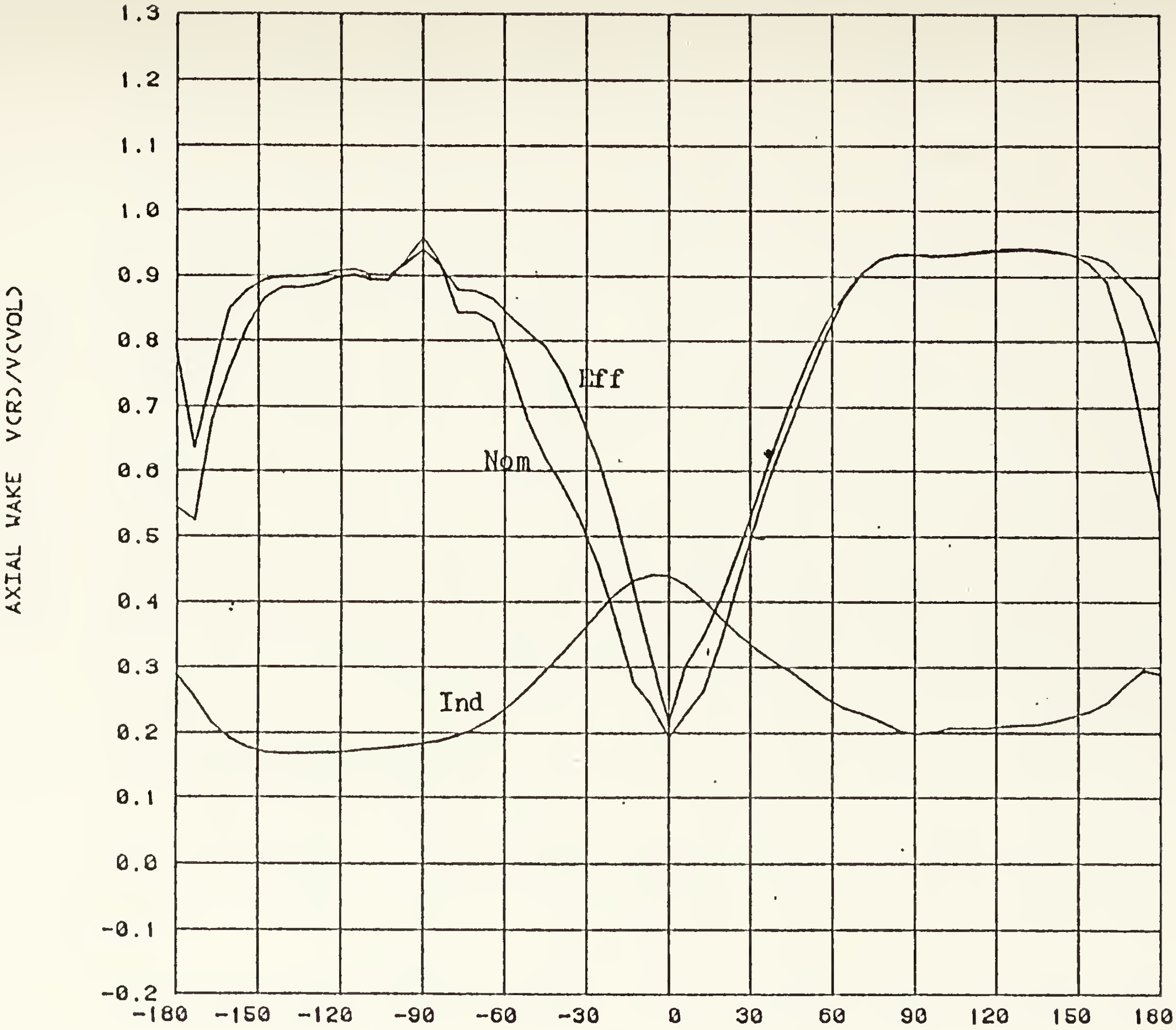
** APPENDIX I **

Nominal, Induced, and Effective Velocities



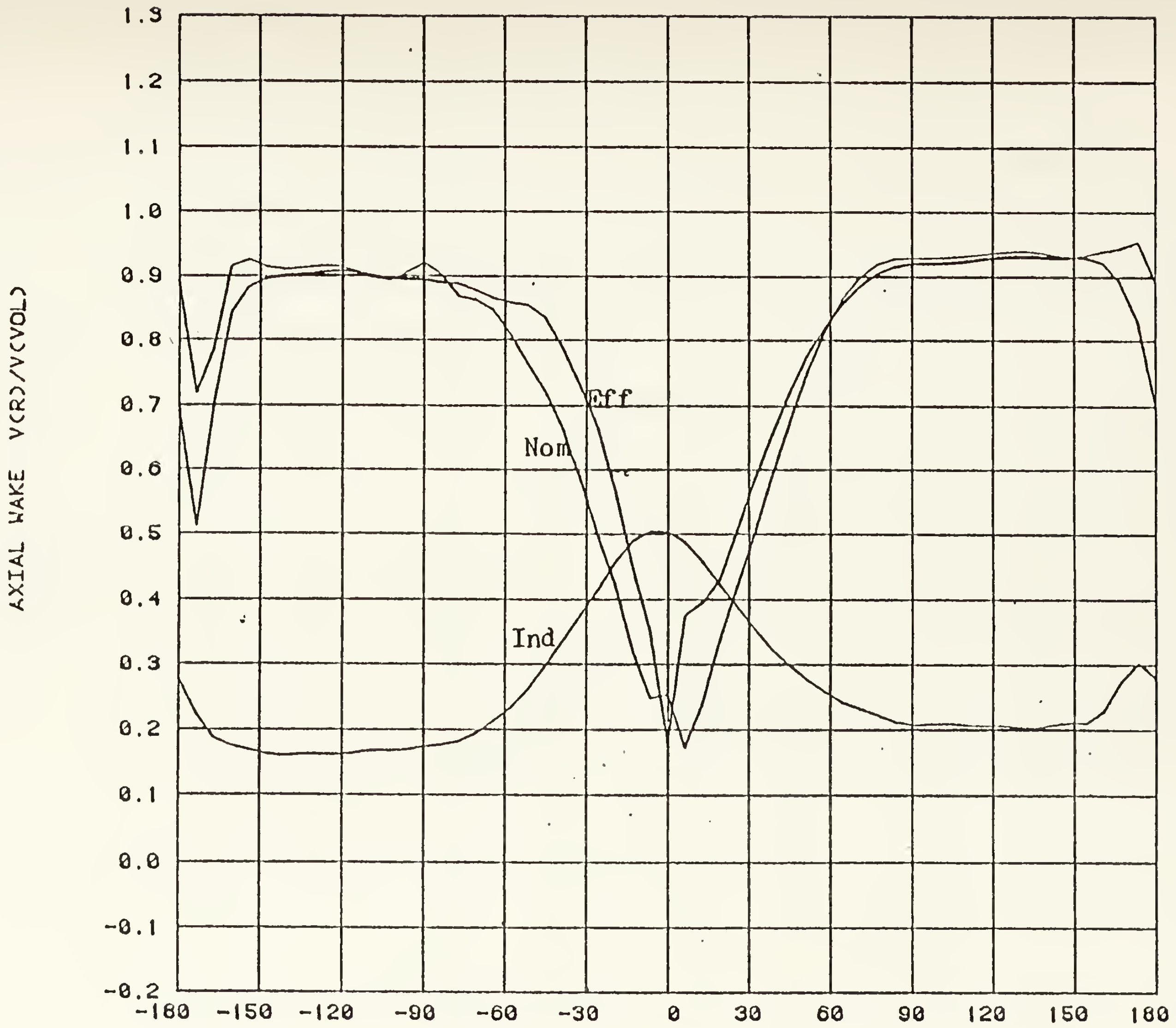
NOM, IND, EFF VELOCITIES, .40 RADIUS

	V(VOL)
Nom	1.0
Ind	1.0
Eff	1.0



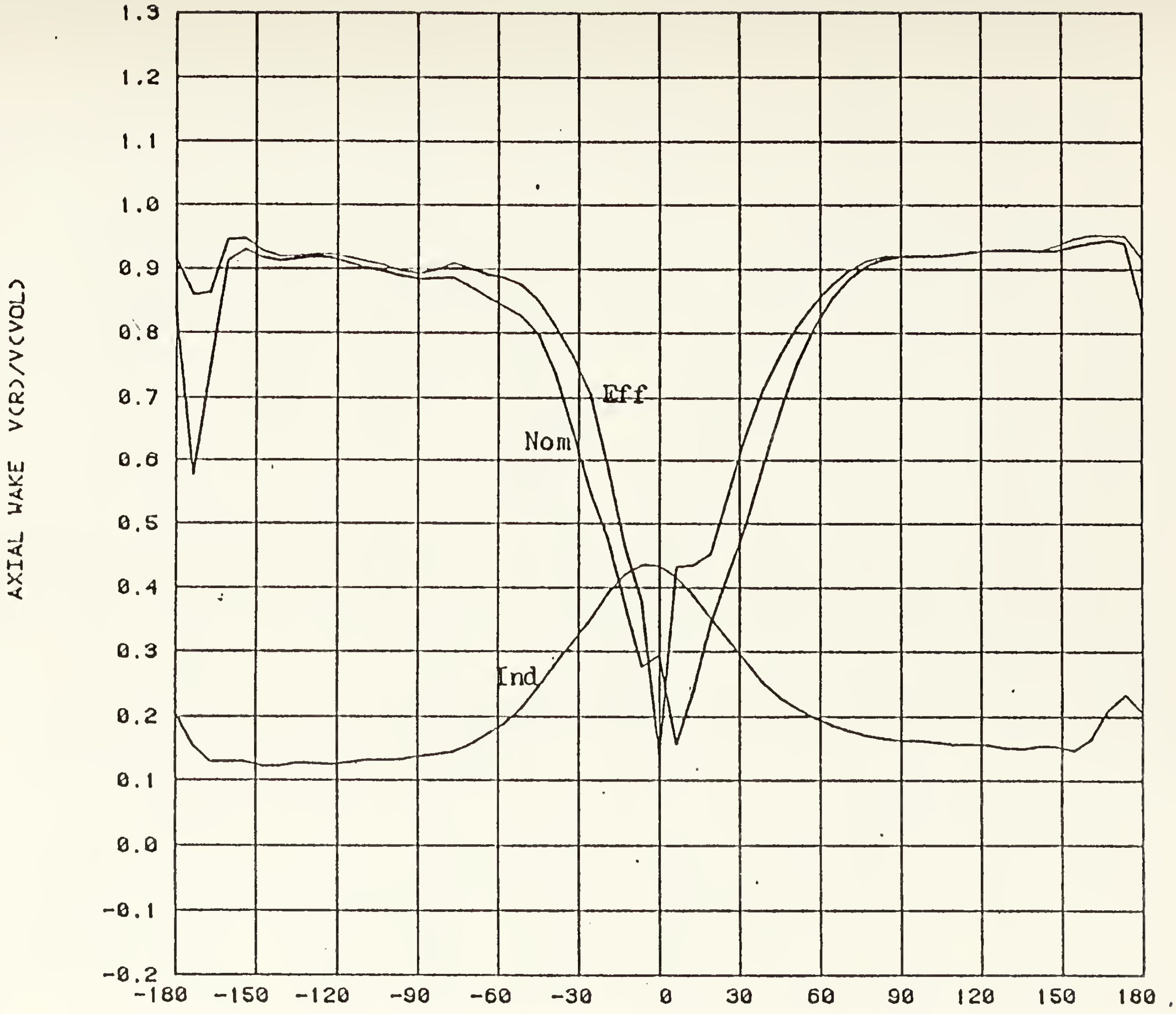
NOM, IND, EFF VELOCITIES, .60 RADIUS

	V(VOL)
Nom	1.0
Ind	1.0
Eff	1.0



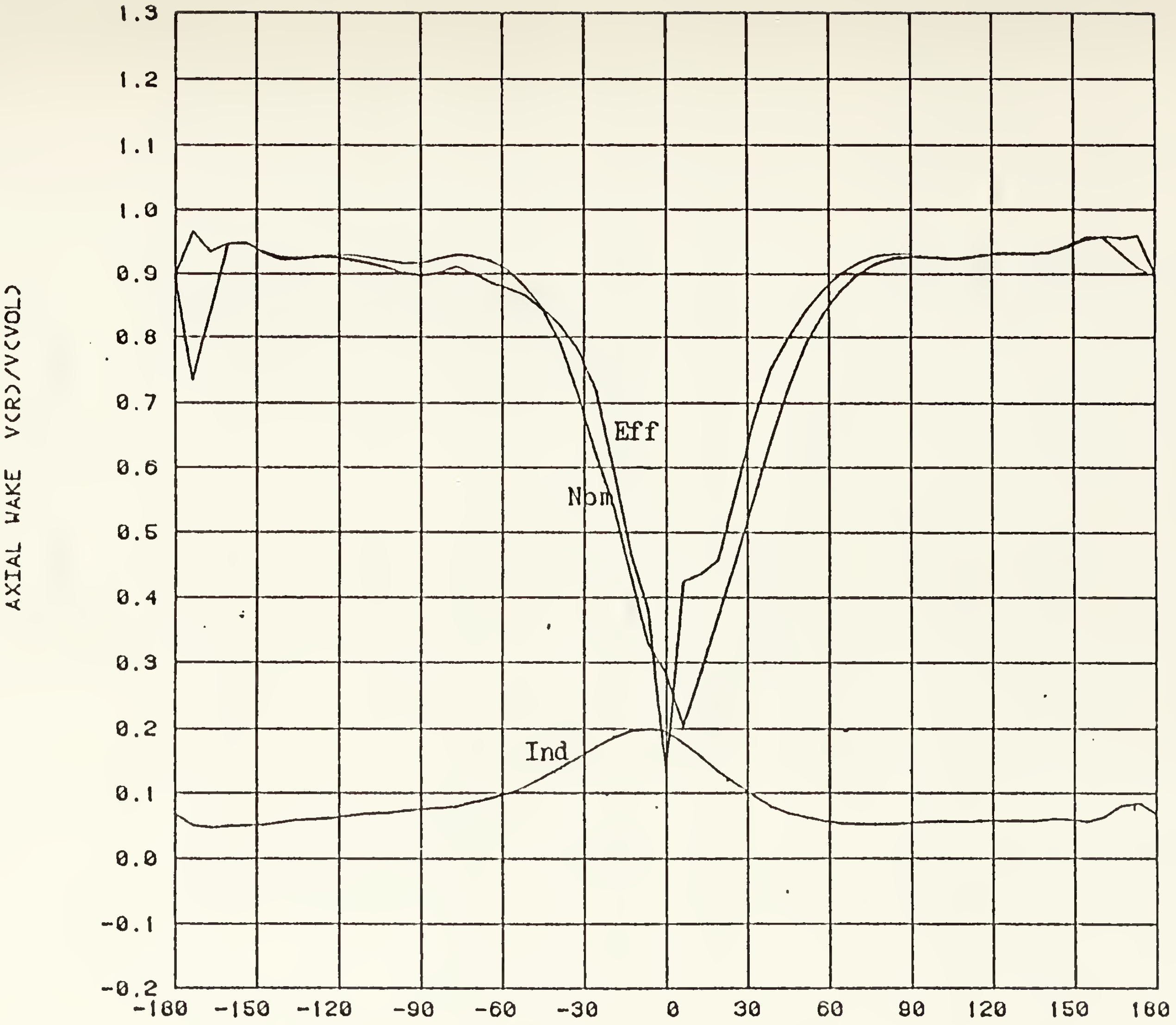
NOM, IND, EFF VELOCITIES, .70 RADIUS

	V(VOL)
Nom	1.0
Ind	1.0
Eff	1.0



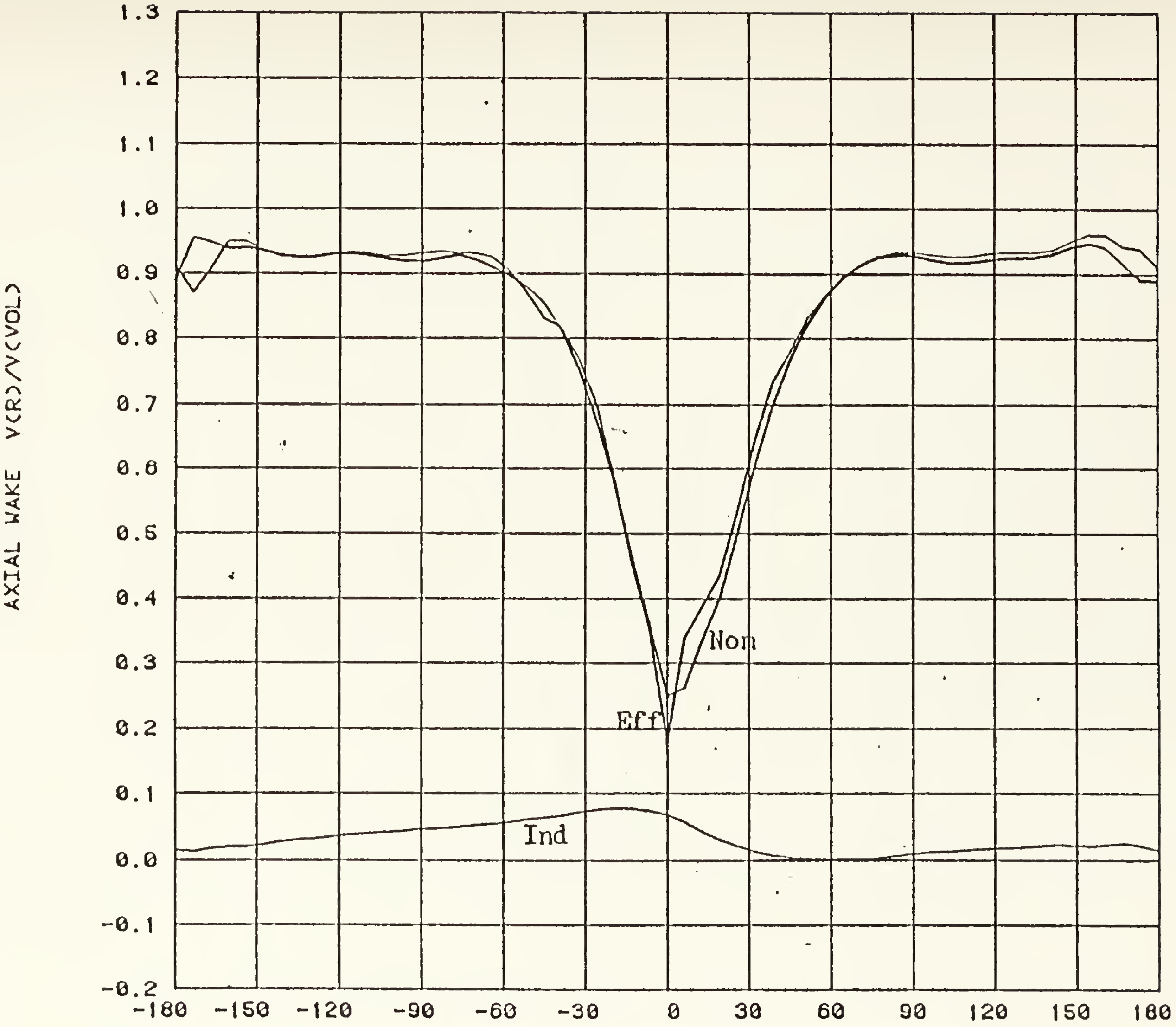
NOM, IND, EFF VELOCITIES, .80 RADIUS

	V(VOL)
Nom	1.0
Ind	1.0
Eff	1.0



NOM, IND, EFF VELOCITIES, .90 RADIUS.

	V(VOL)
Nom	1.0
Ind	1.0
Eff	1.0

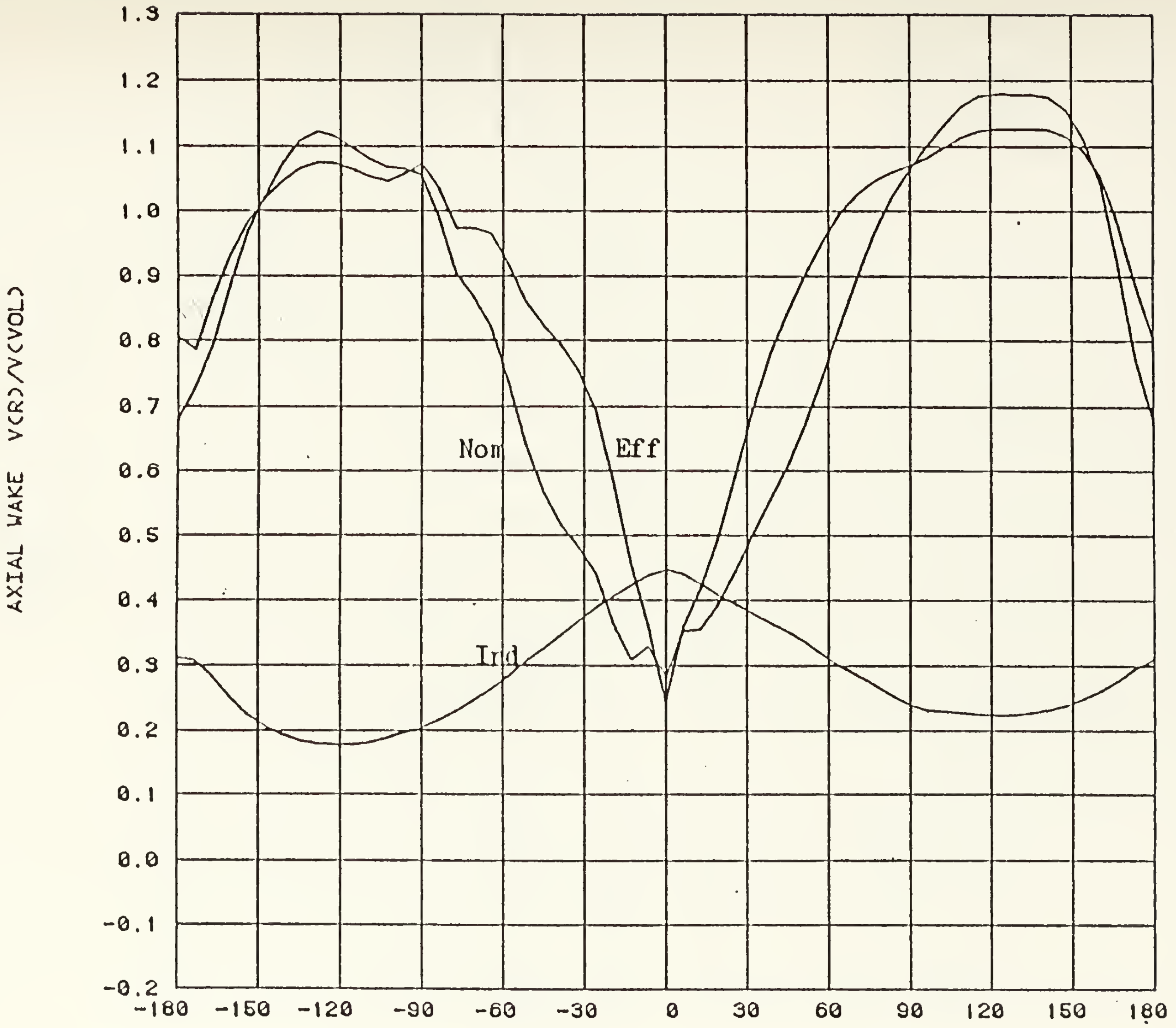


NOM, IND, EFF VELOCITIES, .98 RADIUS

** APPENDIX II **

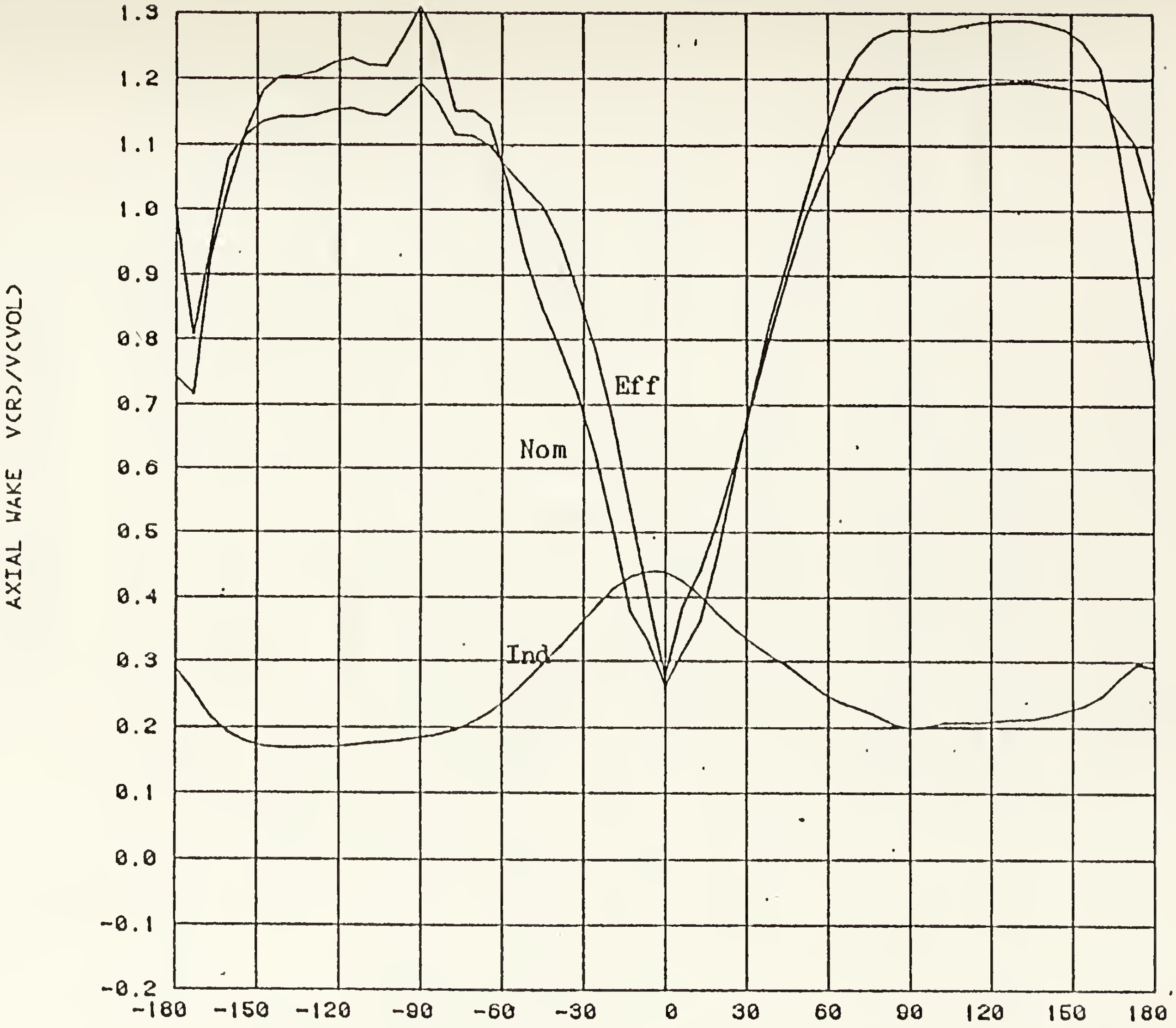
Nominal, Induced, and Effective Velocities, Non-Dimensionalized

	V(VOL)
Nom	0.732
Ind	1.0
Eff	0.787



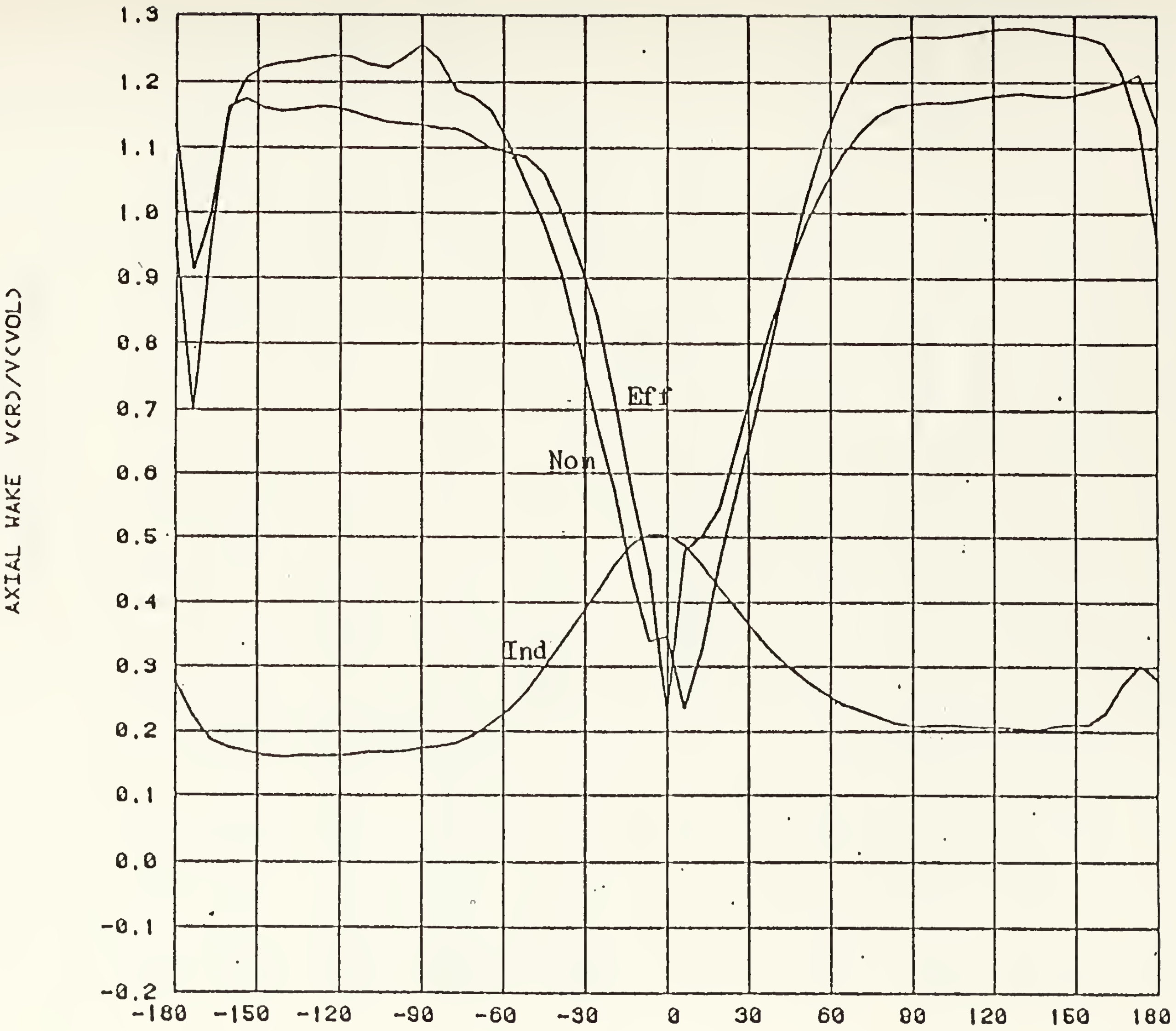
NOM, IND, EFF VELOCITIES, .40 RADIUS

	V(VOL)
Nom	0.732
Ind	1.0
Eff	0.787

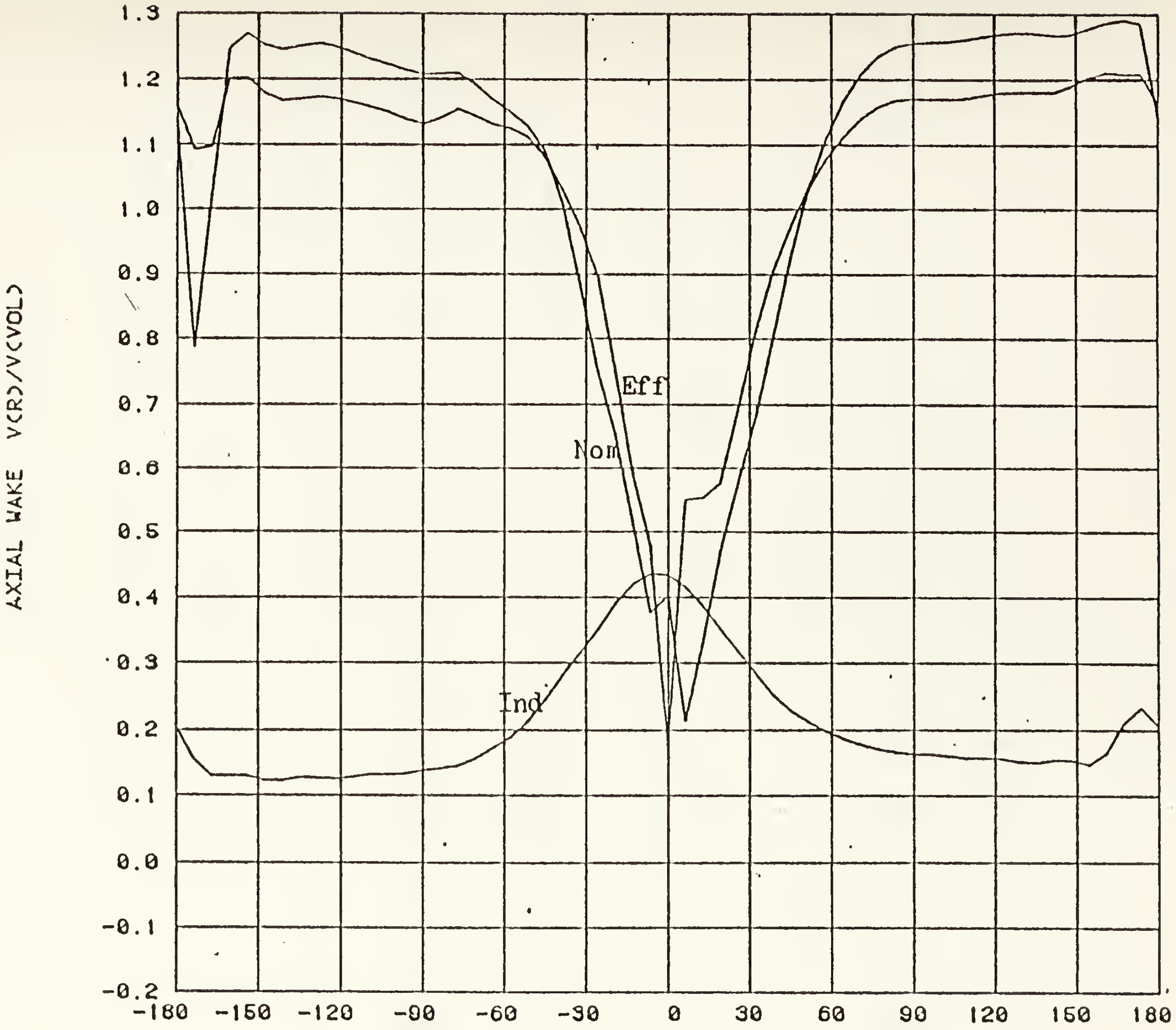


NOM, IND, EFF VELOCITIES, .60 RADIUS

	V(VOL)
Nom	0.732
Ind	1.0
Eff	0.787



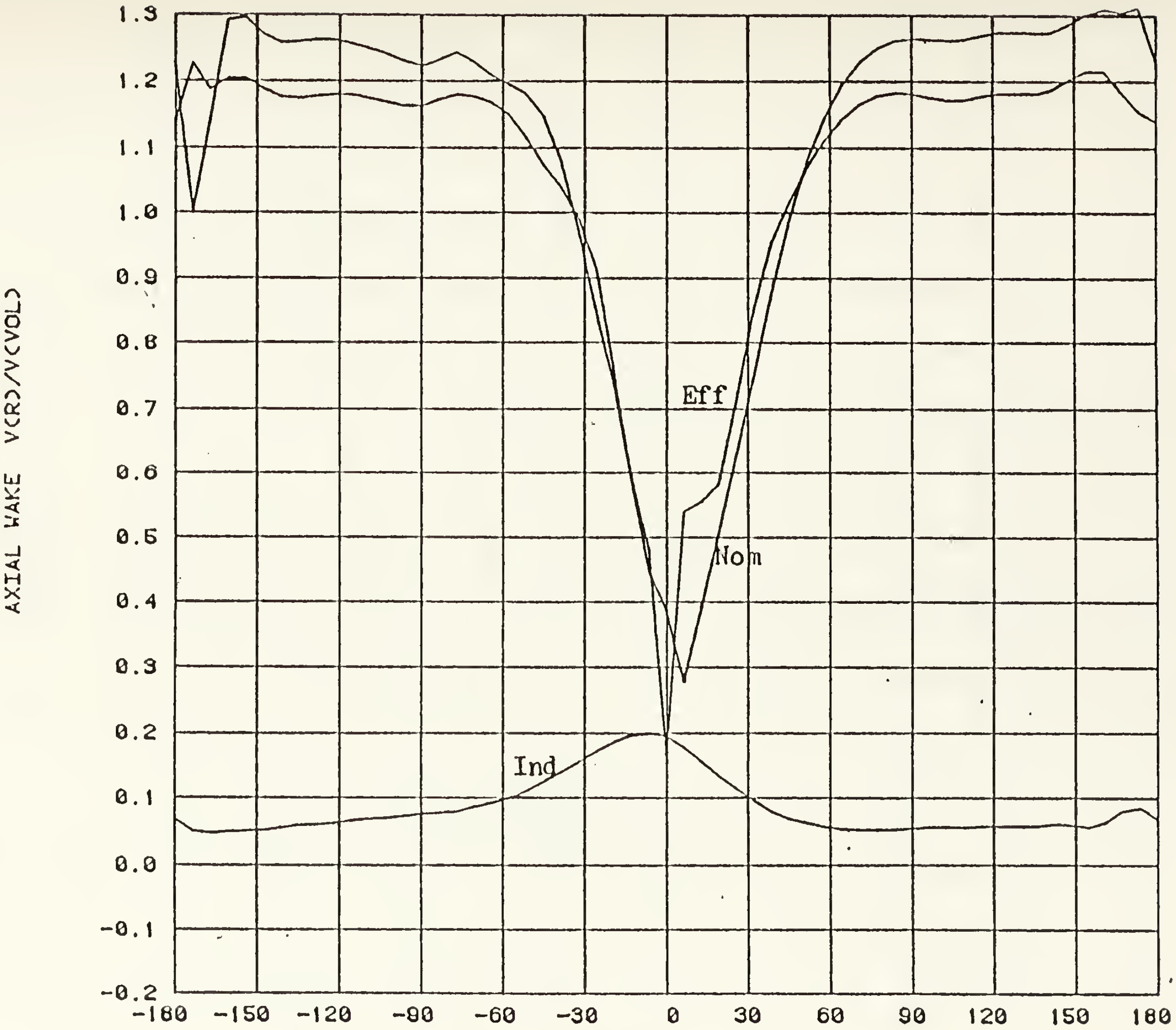
NOM, IND, EFF VELOCITIES, .70 RADIUS



NOM, IND, EFF VELOCITIES, .80 RADIUS

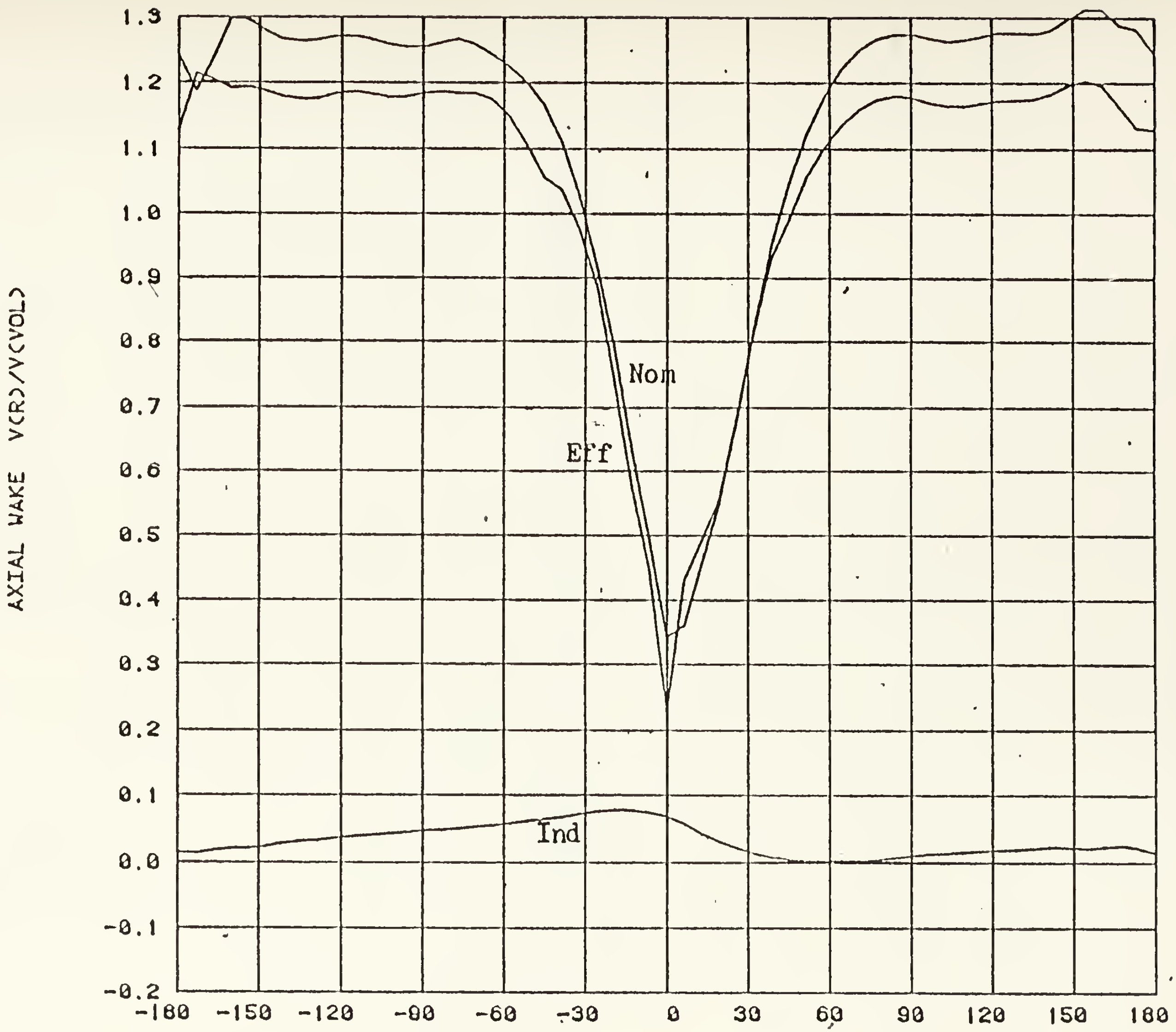
	V(VOL)
Nom	0.732
Ind	1.0
Eff	0.787

	V(VOL)
Nom	0.732
Ind	1.0
Eff	0.787



NOM, IND, EFF VELOCITIES, .90 RADIUS

	V(VOL)
Nom	0.732
Ind	1.0
Eff	0.787

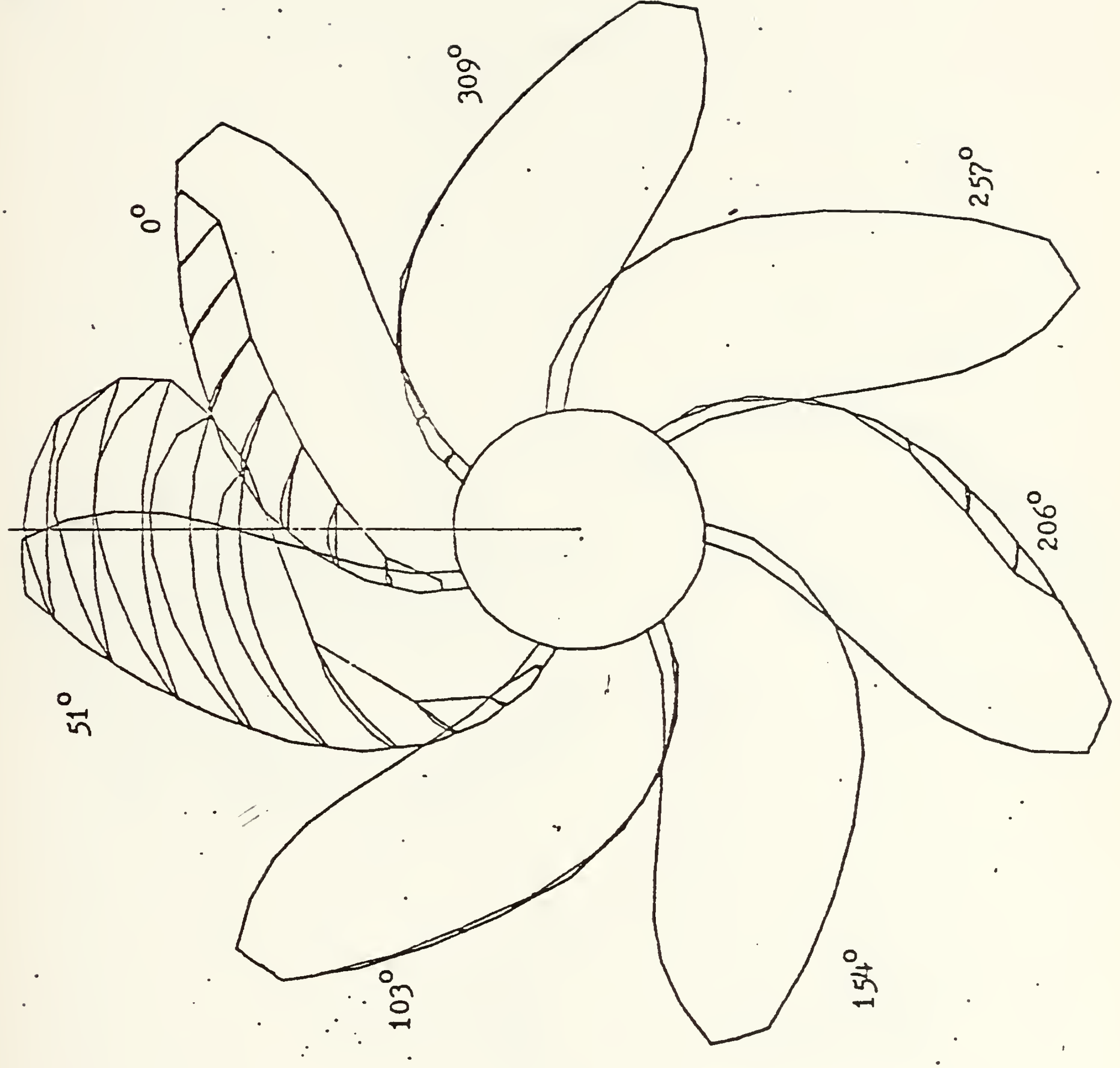


NOM, IND, EFF VELOCITIES, .98 RADIUS

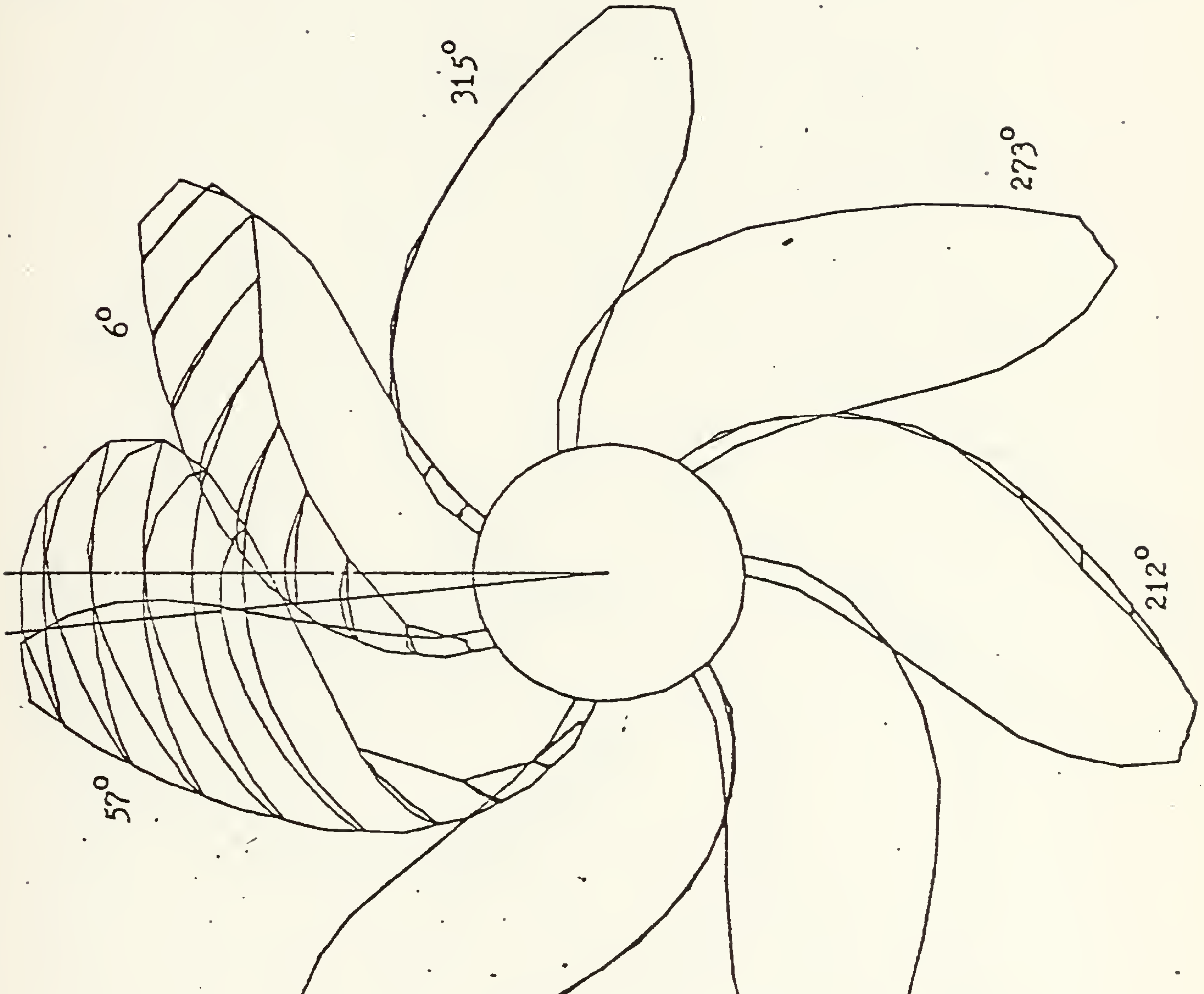
** APPENDIX III **

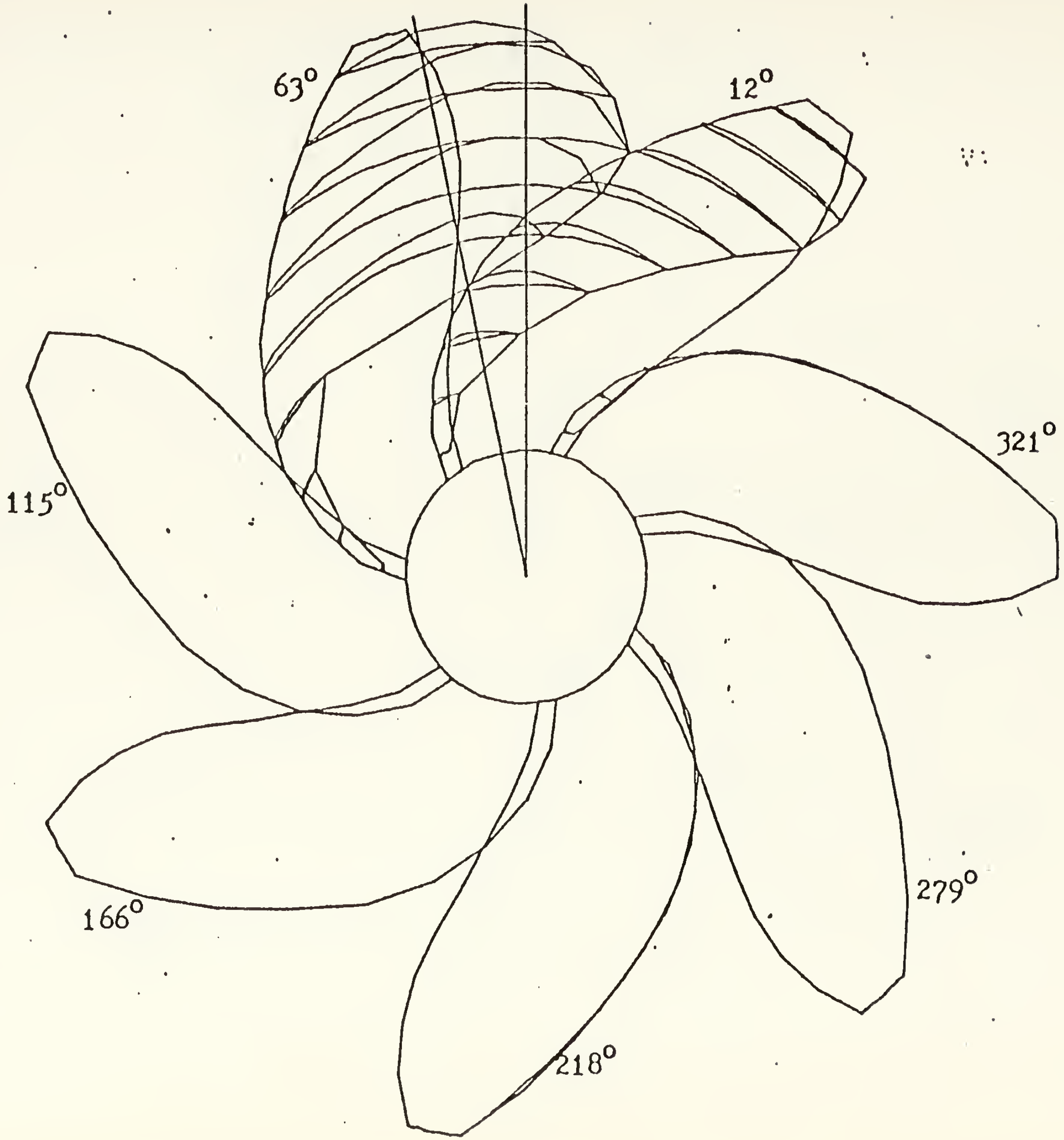
Propeller Cavitation, Nominal Wake

Nominal Wake .0°



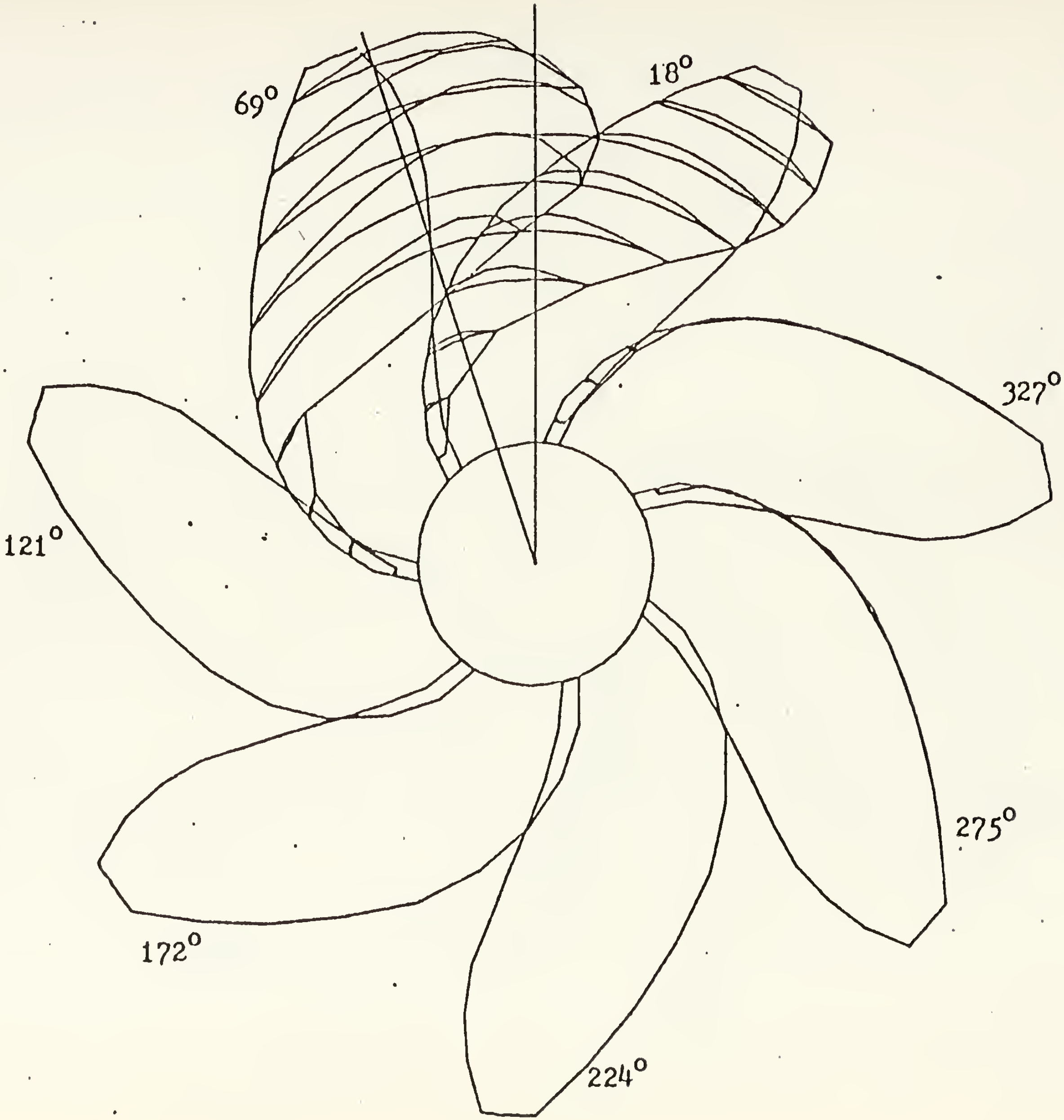
Nominal Wake
6°

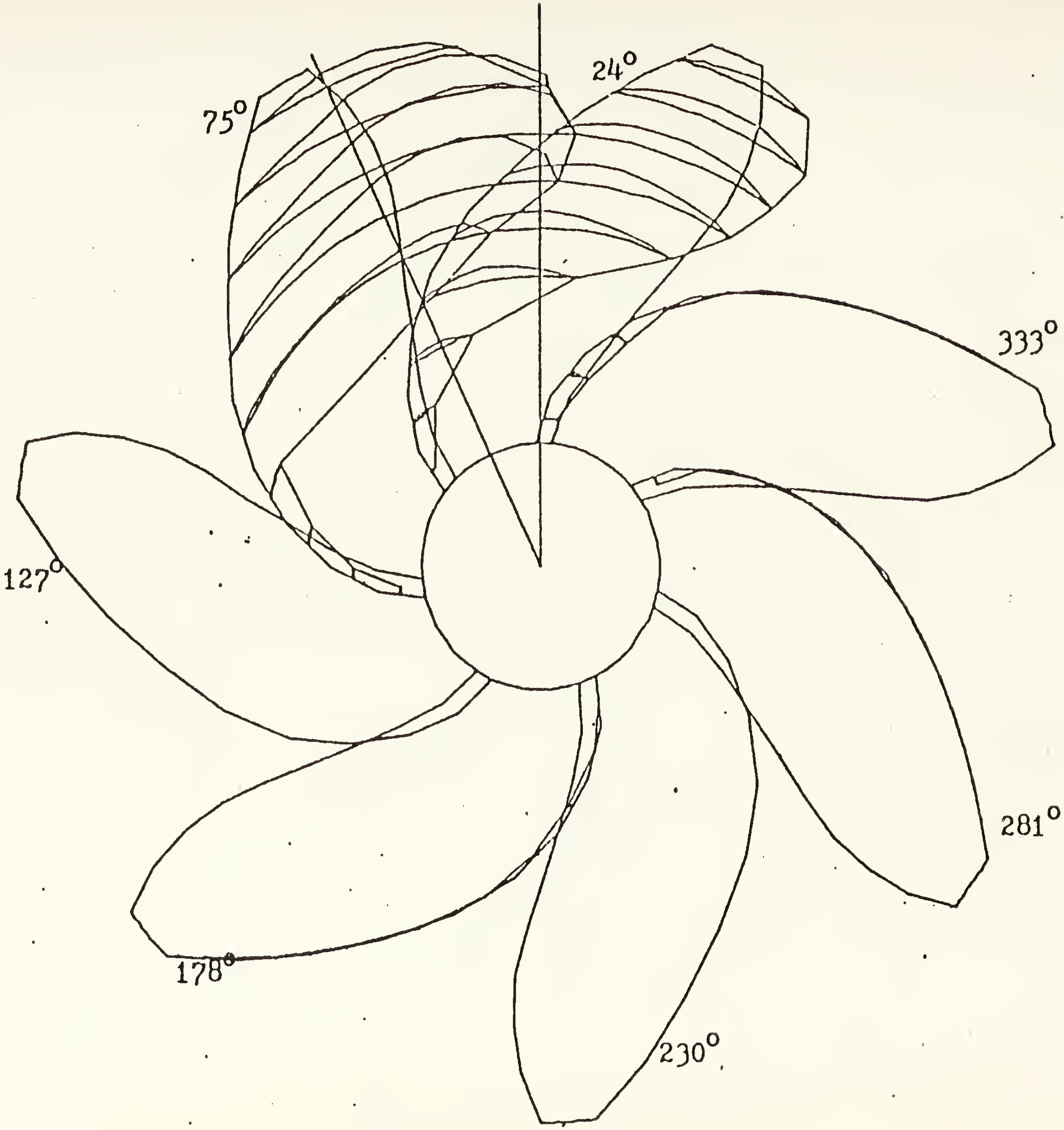




Nominal Wake
12°

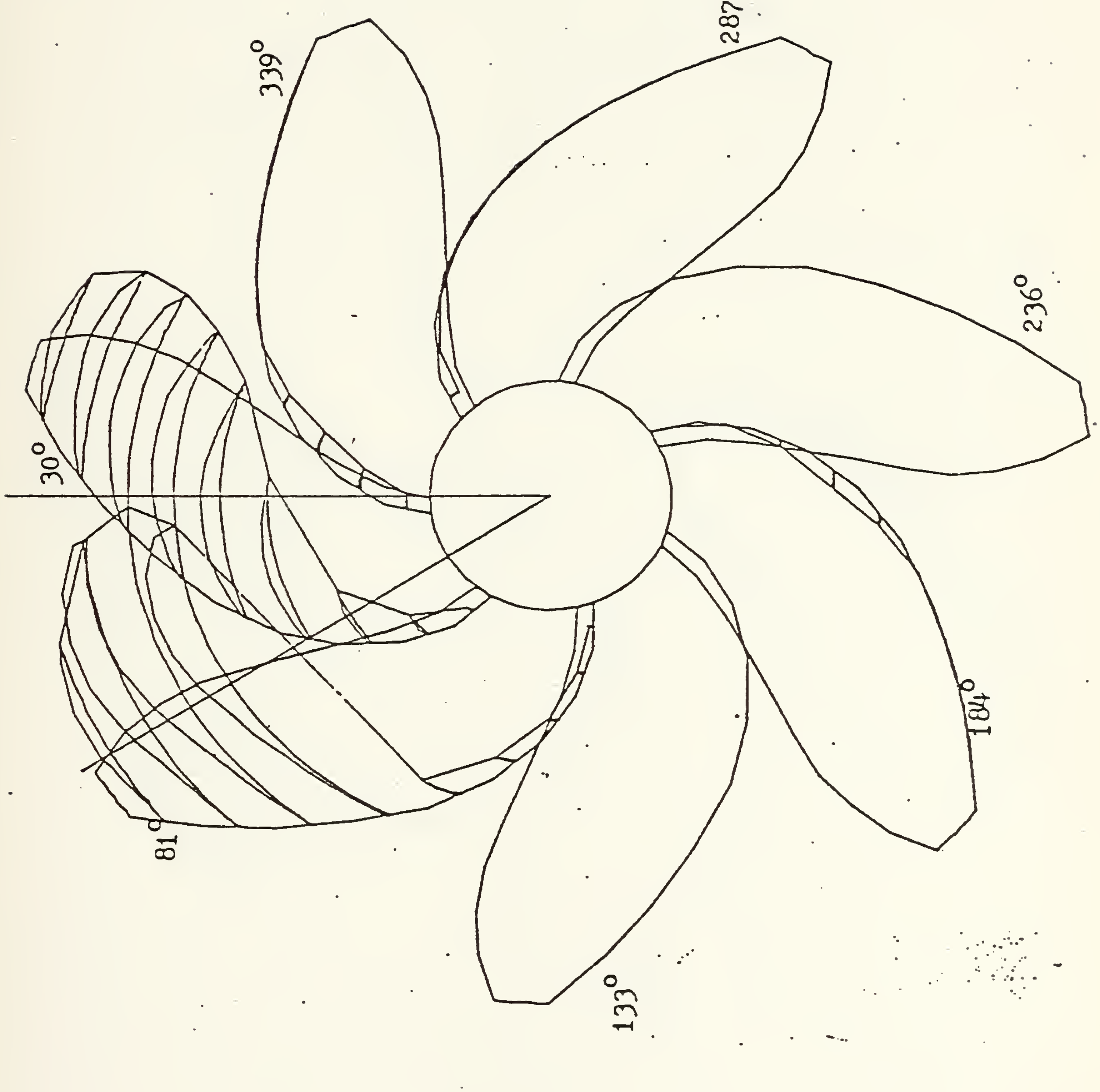
Nominal Wake
18°

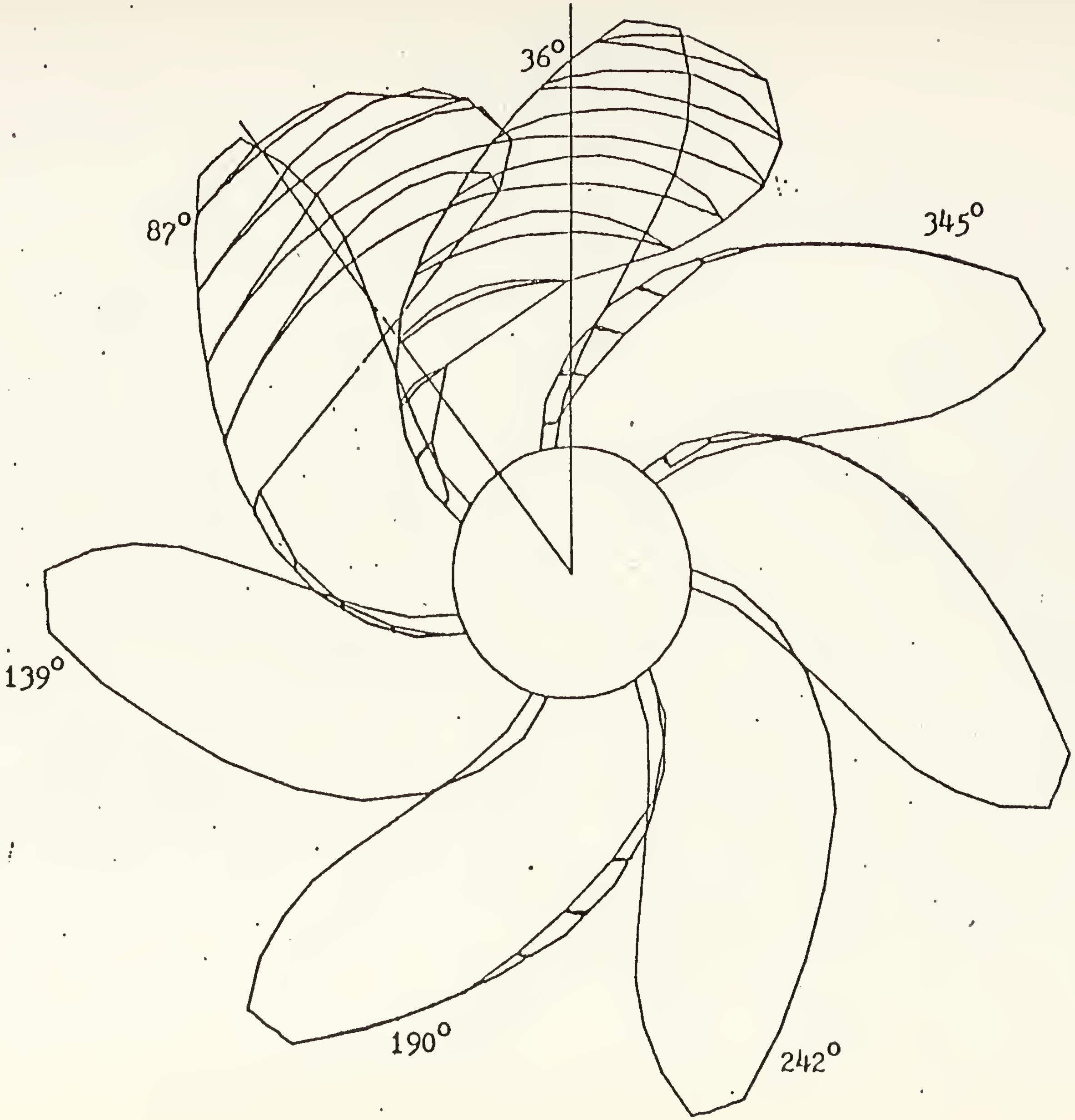




Nominal Wake
24°

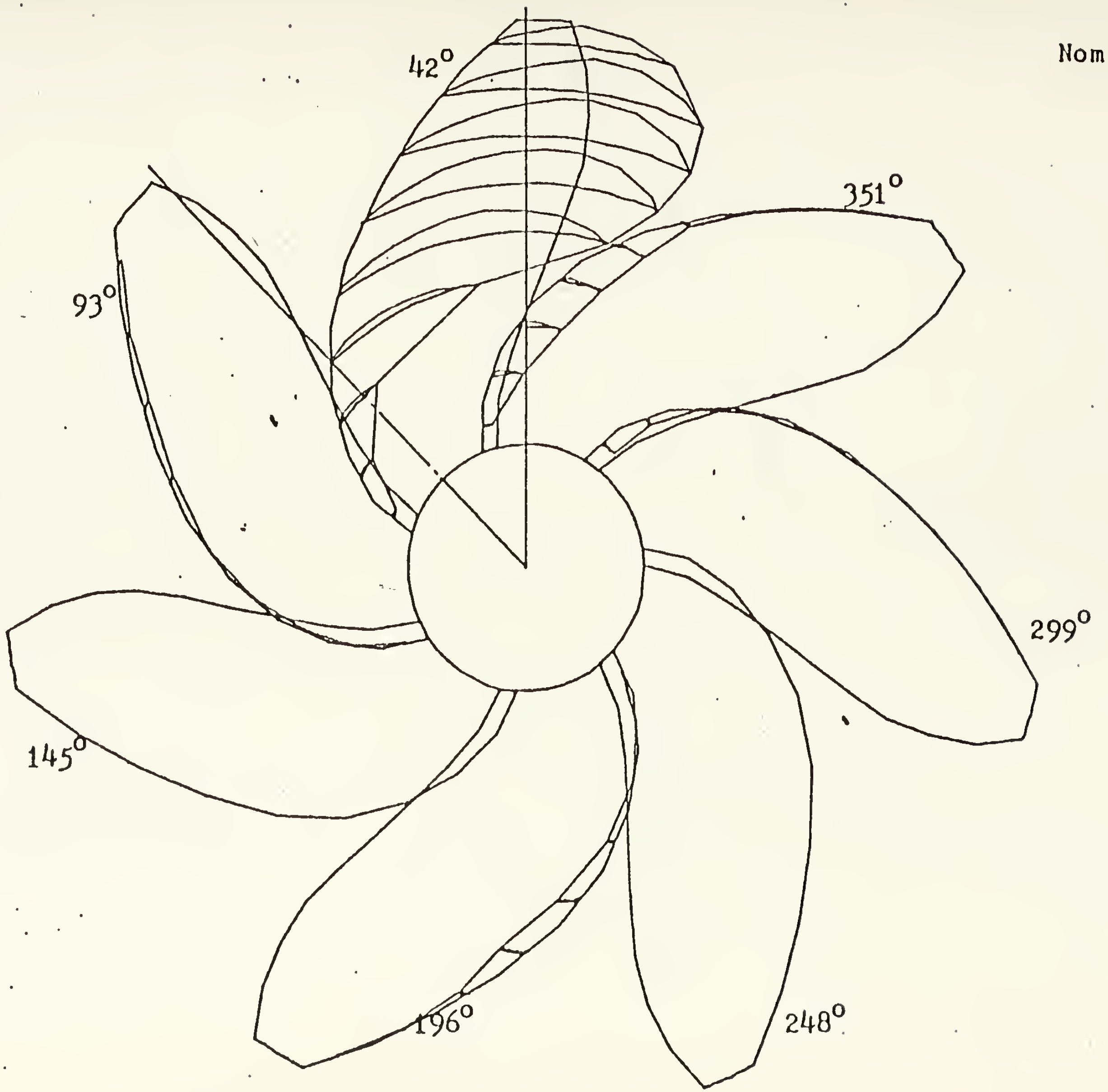
Nominal Wake
30^{o}





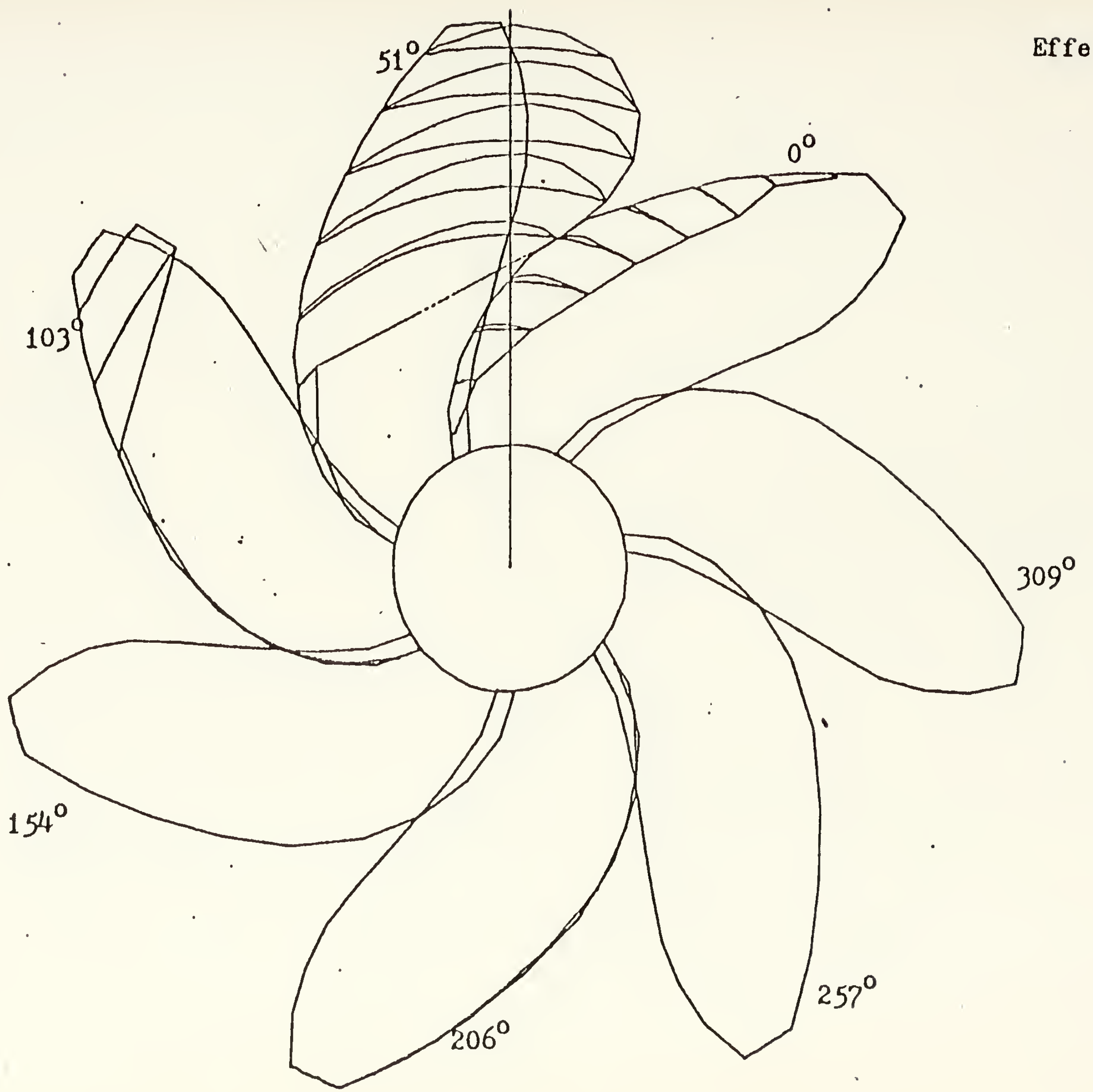
Nominal Wake
36°

Nominal Wake
42°



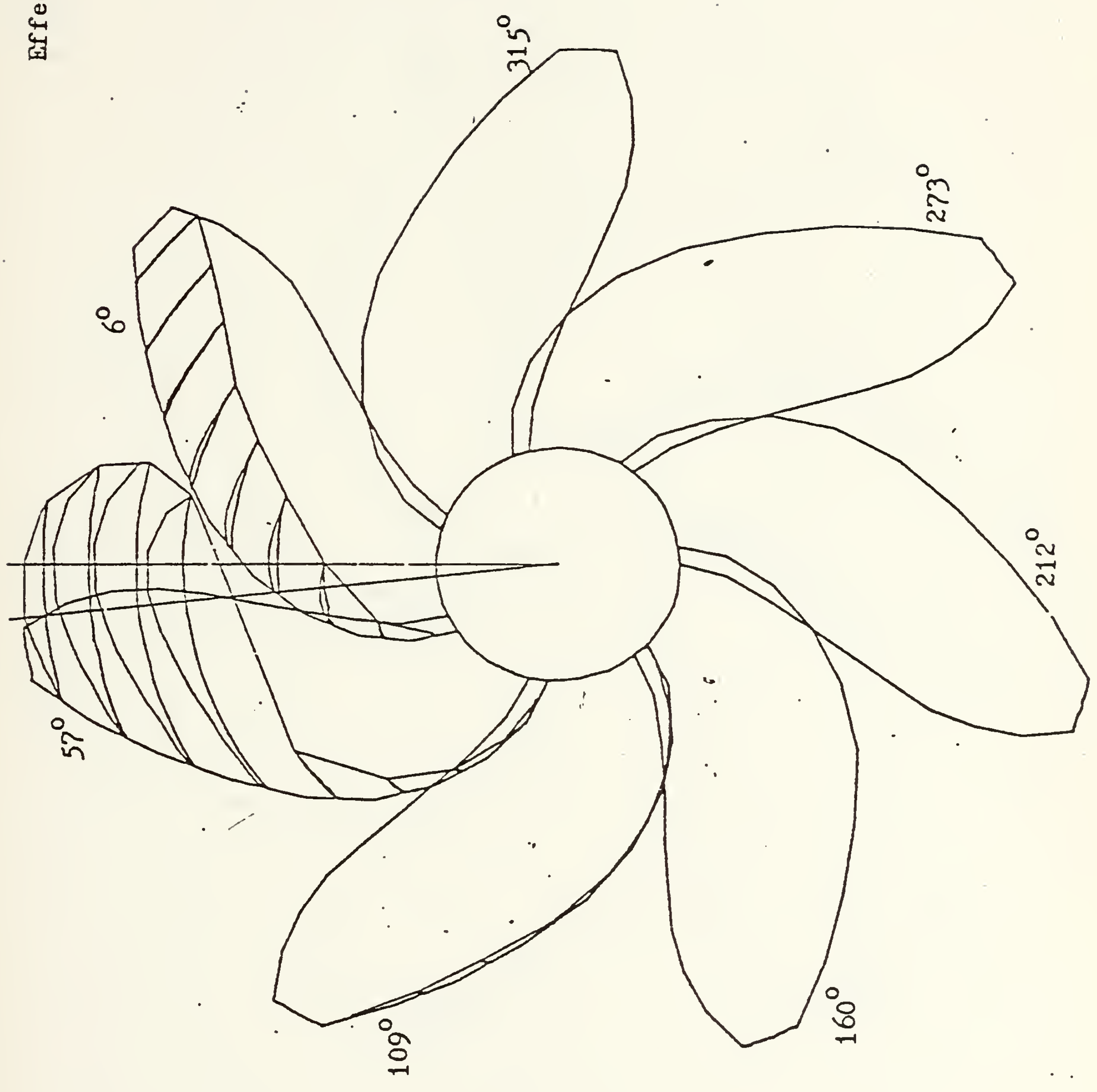
** APPENDIX IV **

Propeller Cavitation, Effective Wake

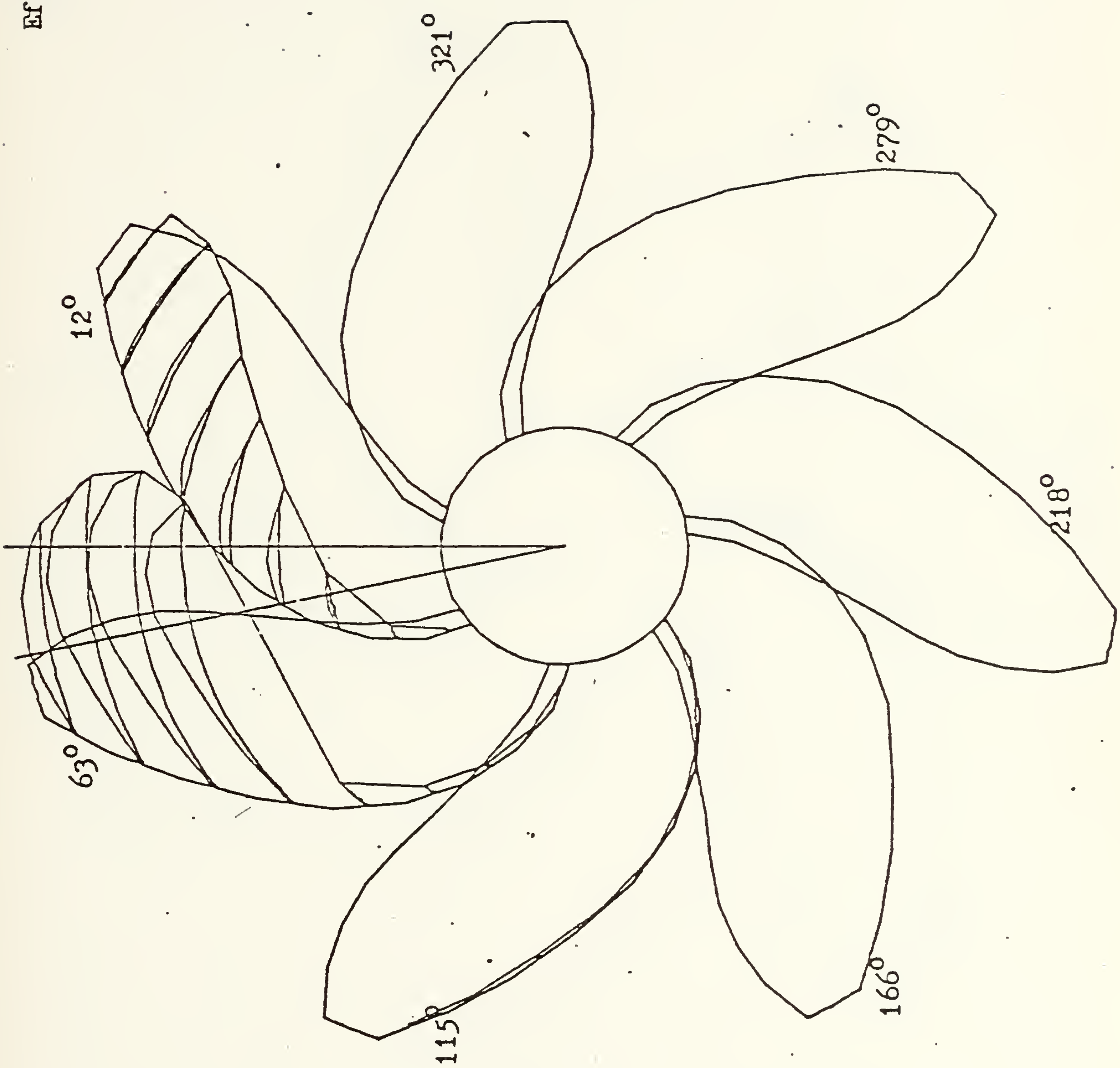


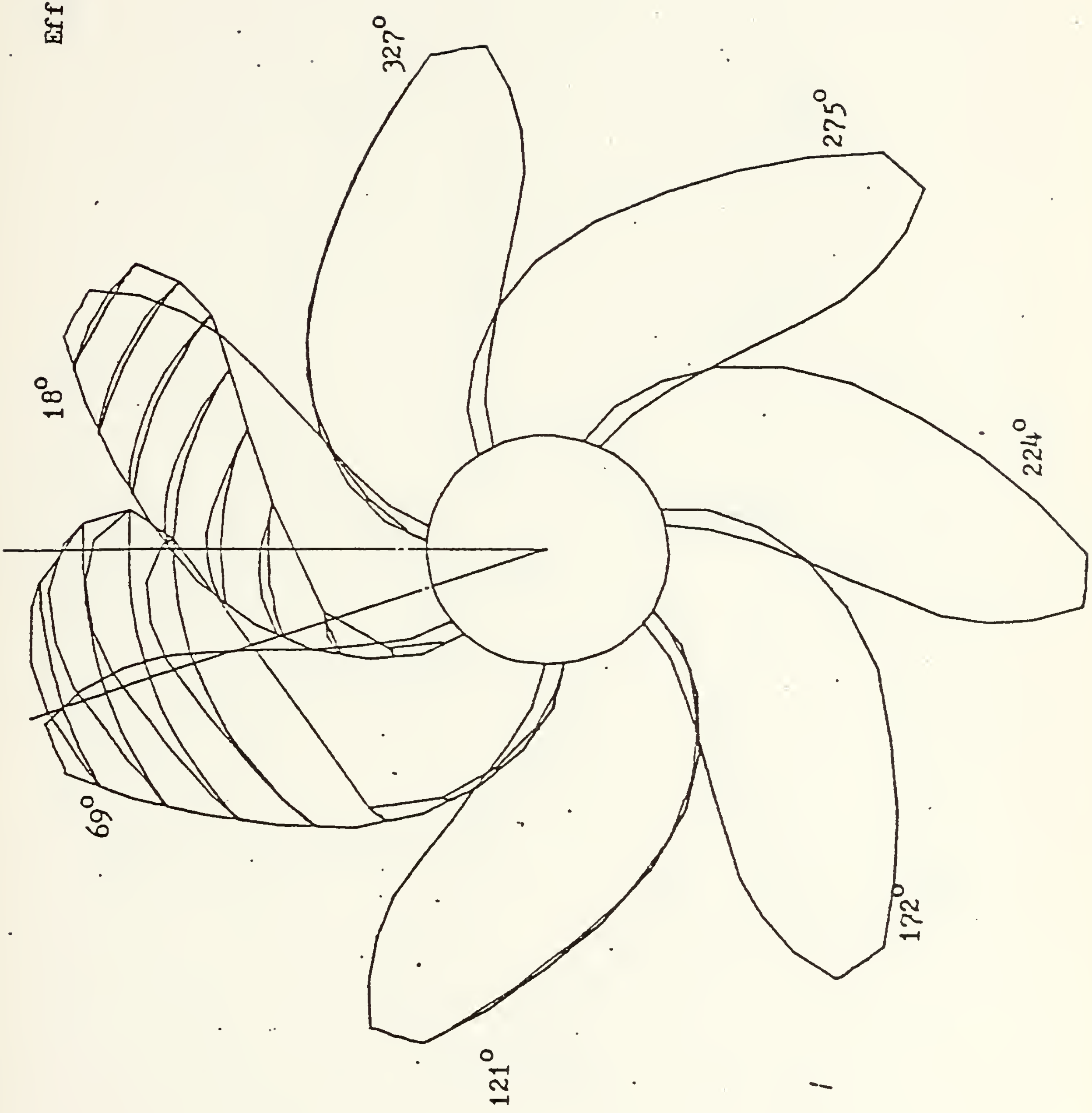
Effective Wake
0°

Effective Wake
6°



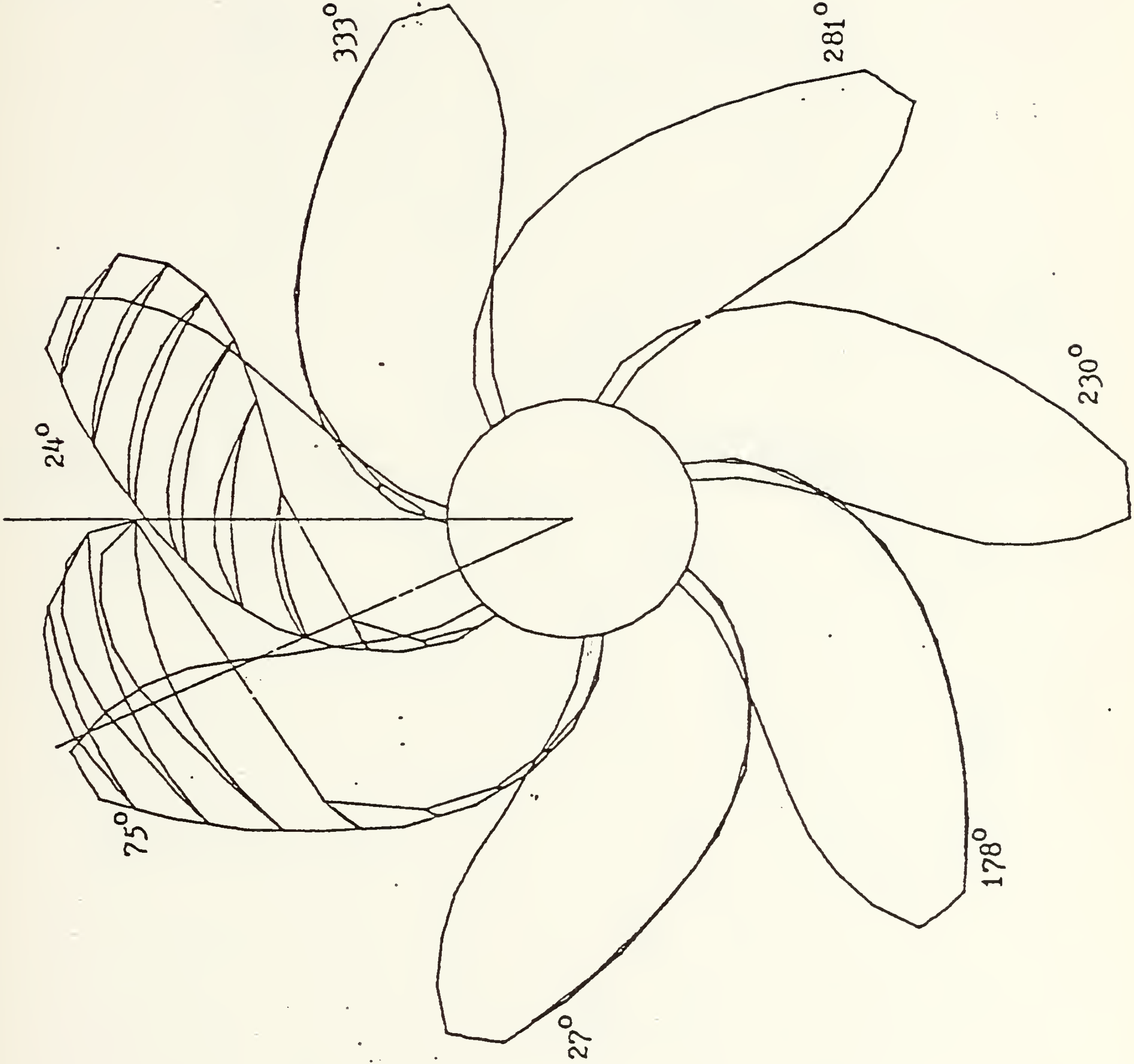
Effective Wake
12°

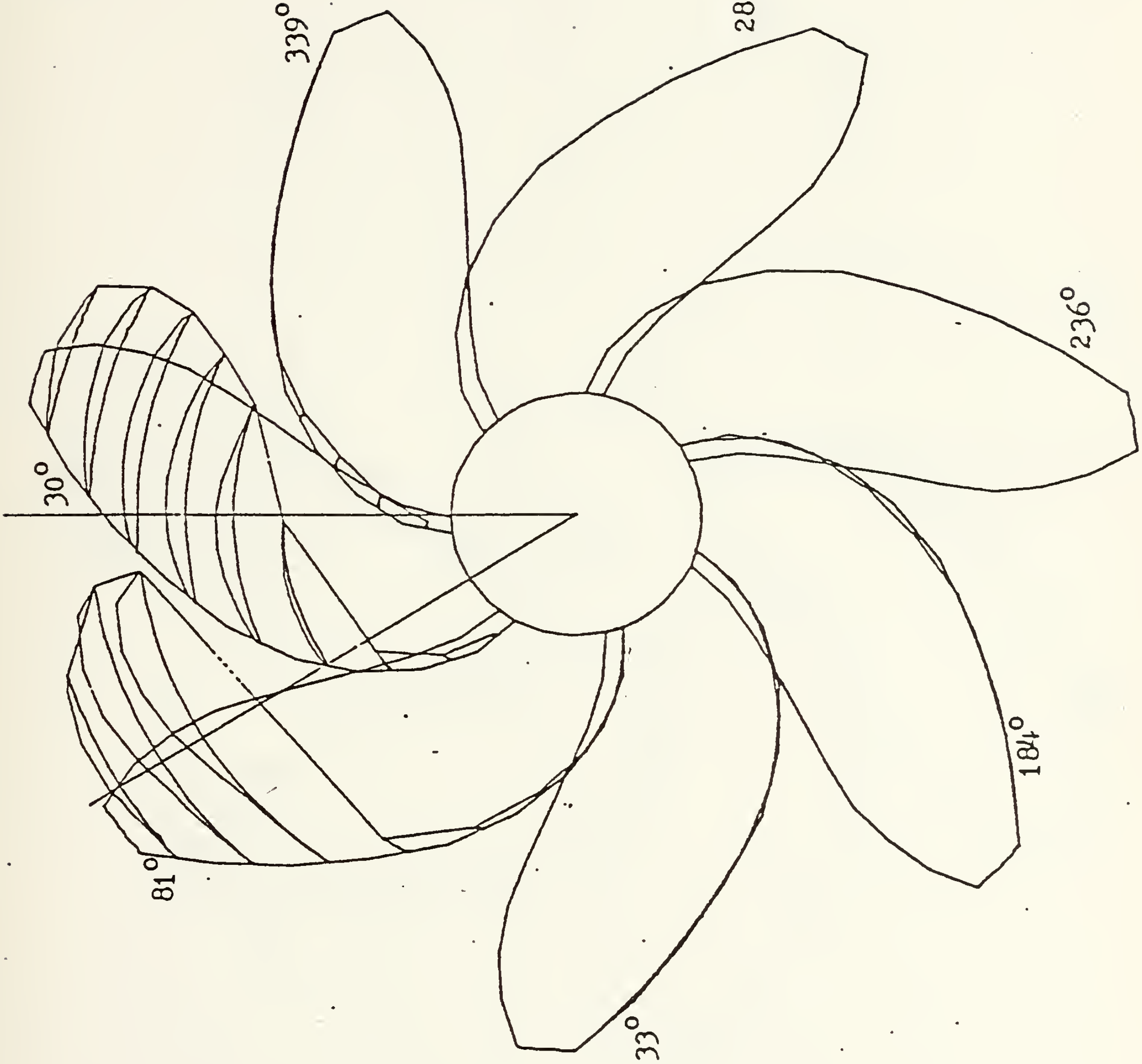




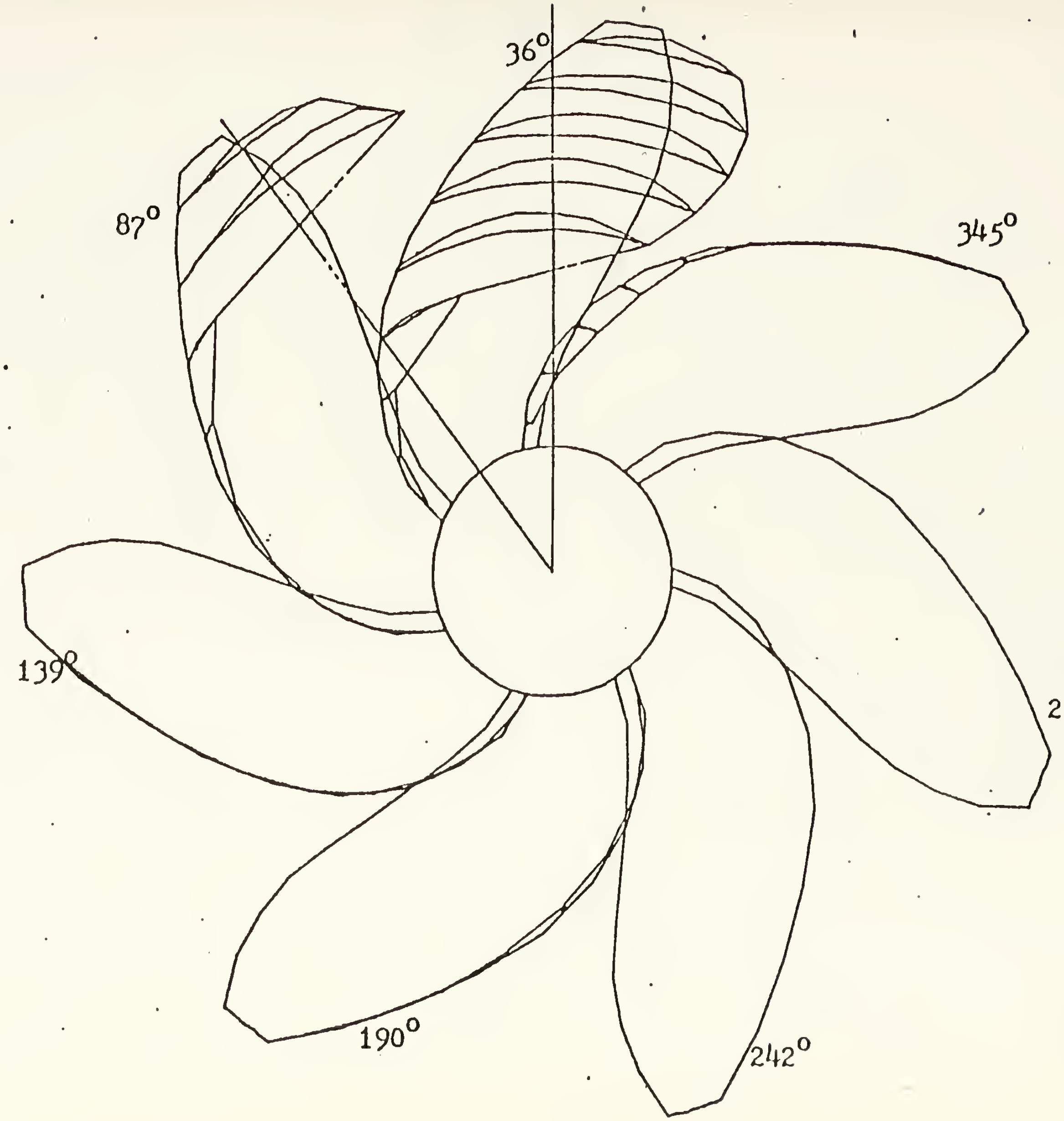
Effective Wake
18°

Effective Wake
24°



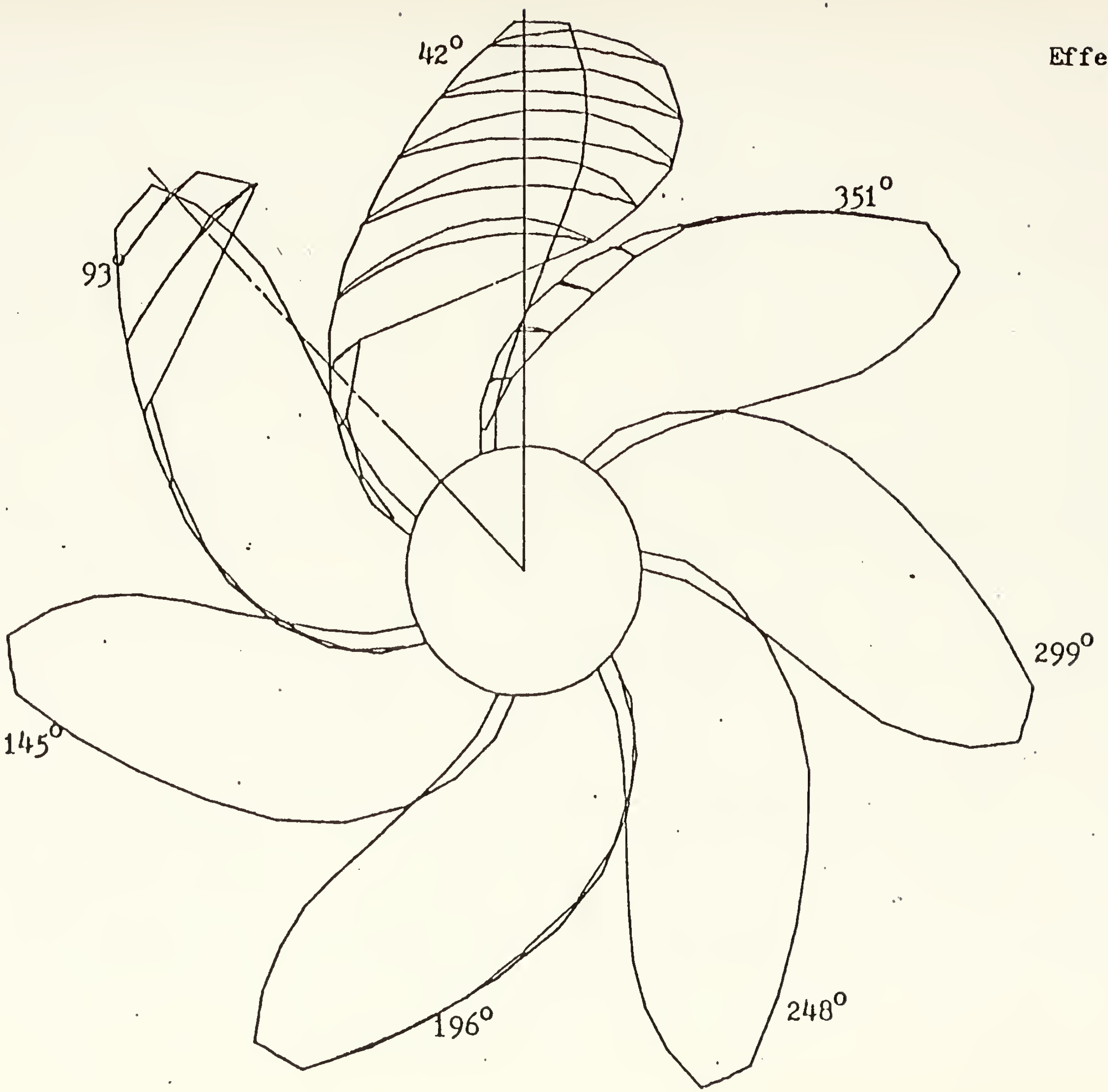


Effective Wake
30°



Effective Wake
36°

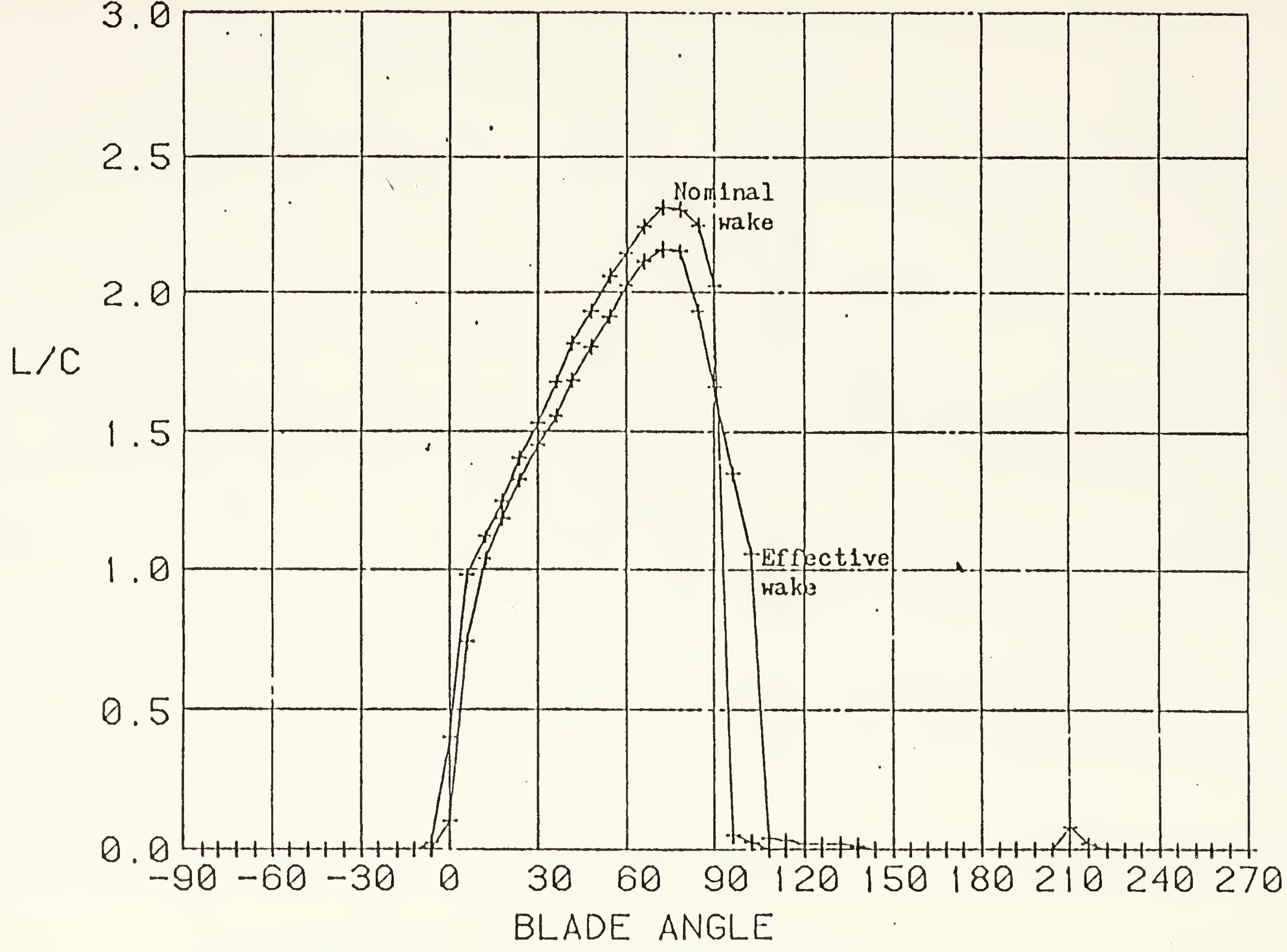
Effective Wake
42°



** APPENDIX V **

Cavity Length, Nominal and Effective Wakes

PUF-3 CAVITY LENGTH OUTPUT

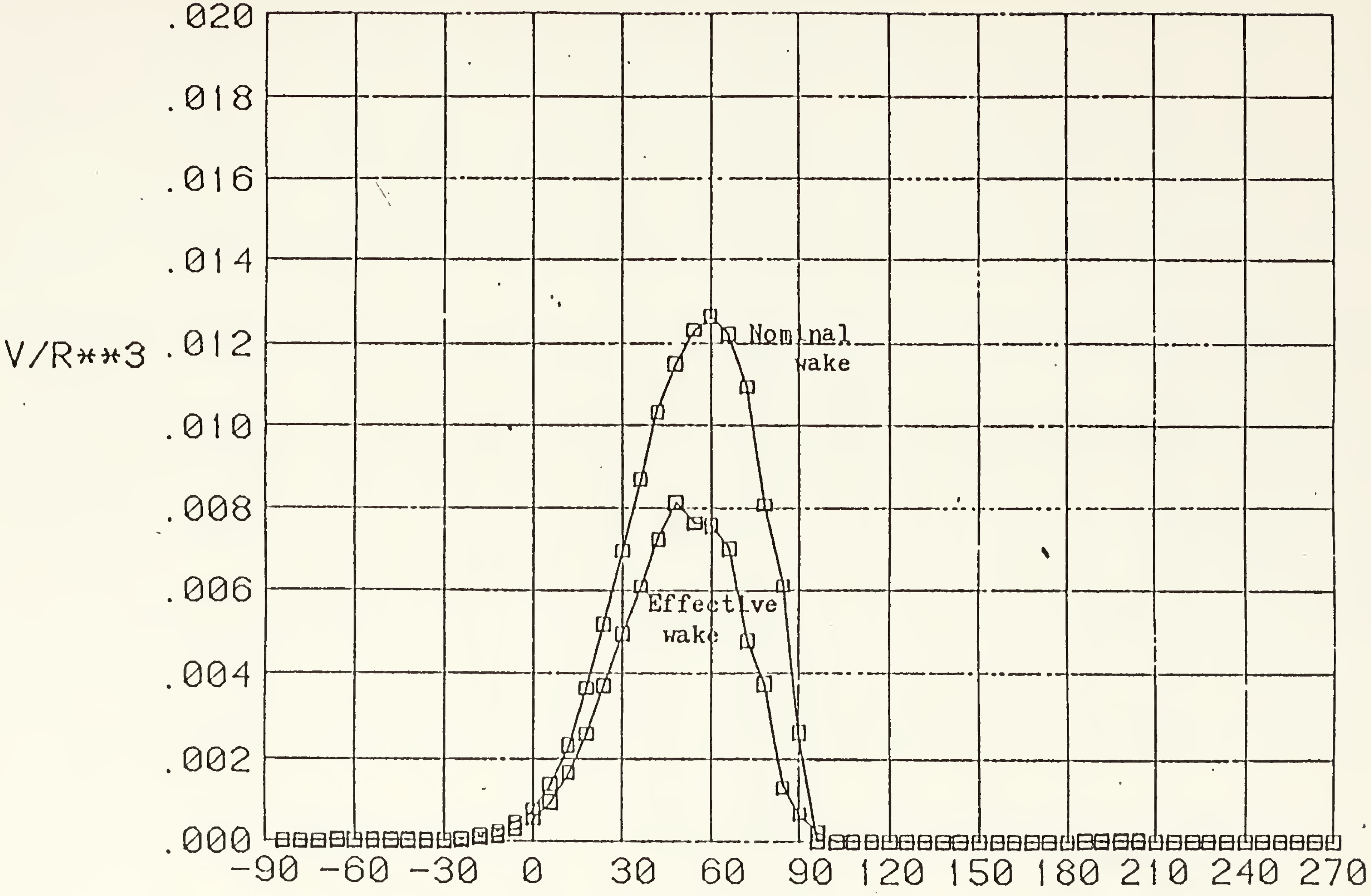


NOMINAL VS. EFFECTIVE WAKE

** APENDIX VI **

Cavity Volume, Nominal and Effective Wakes

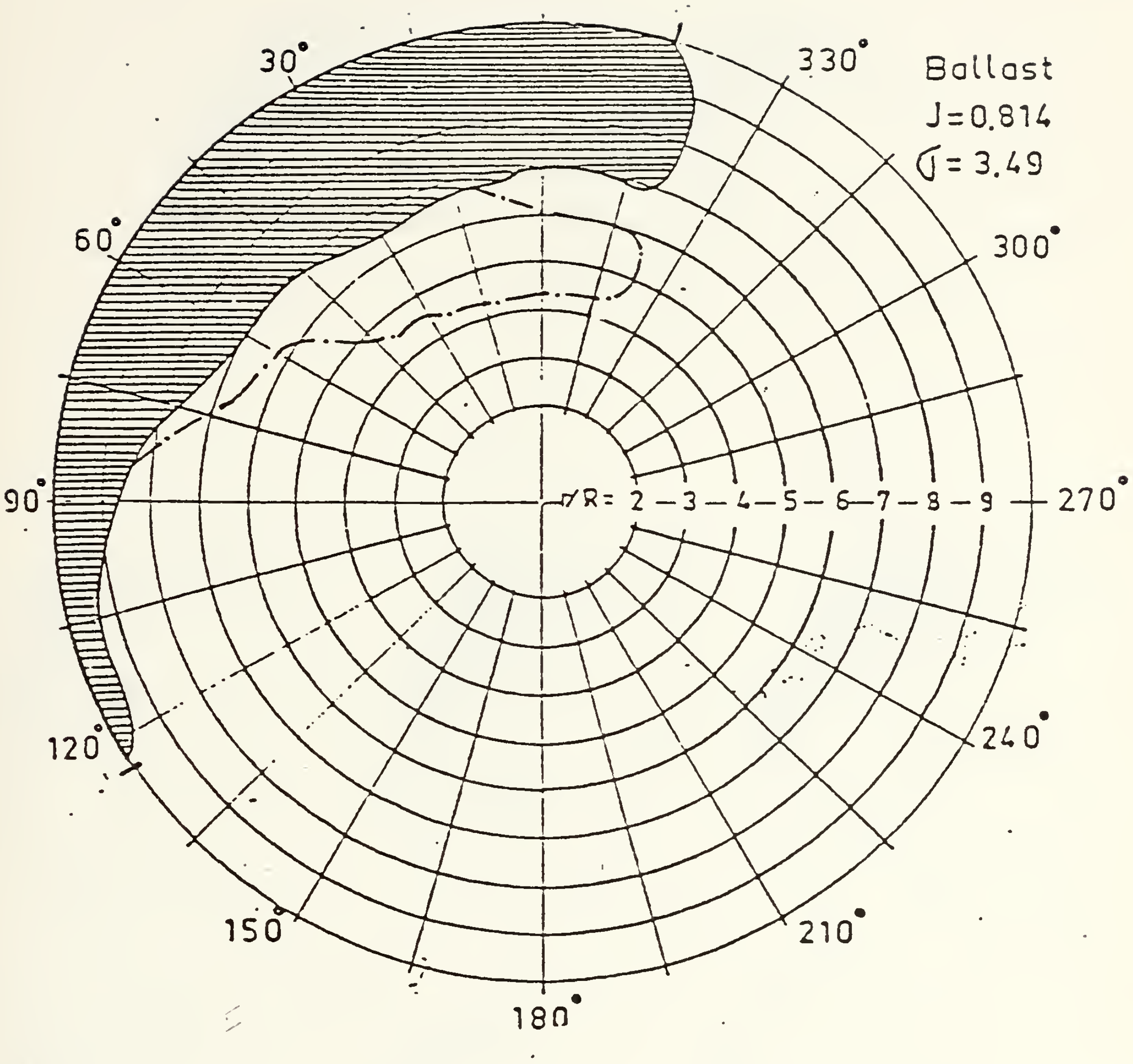
PUF-3 CAVITY VOLUME OUTPUT



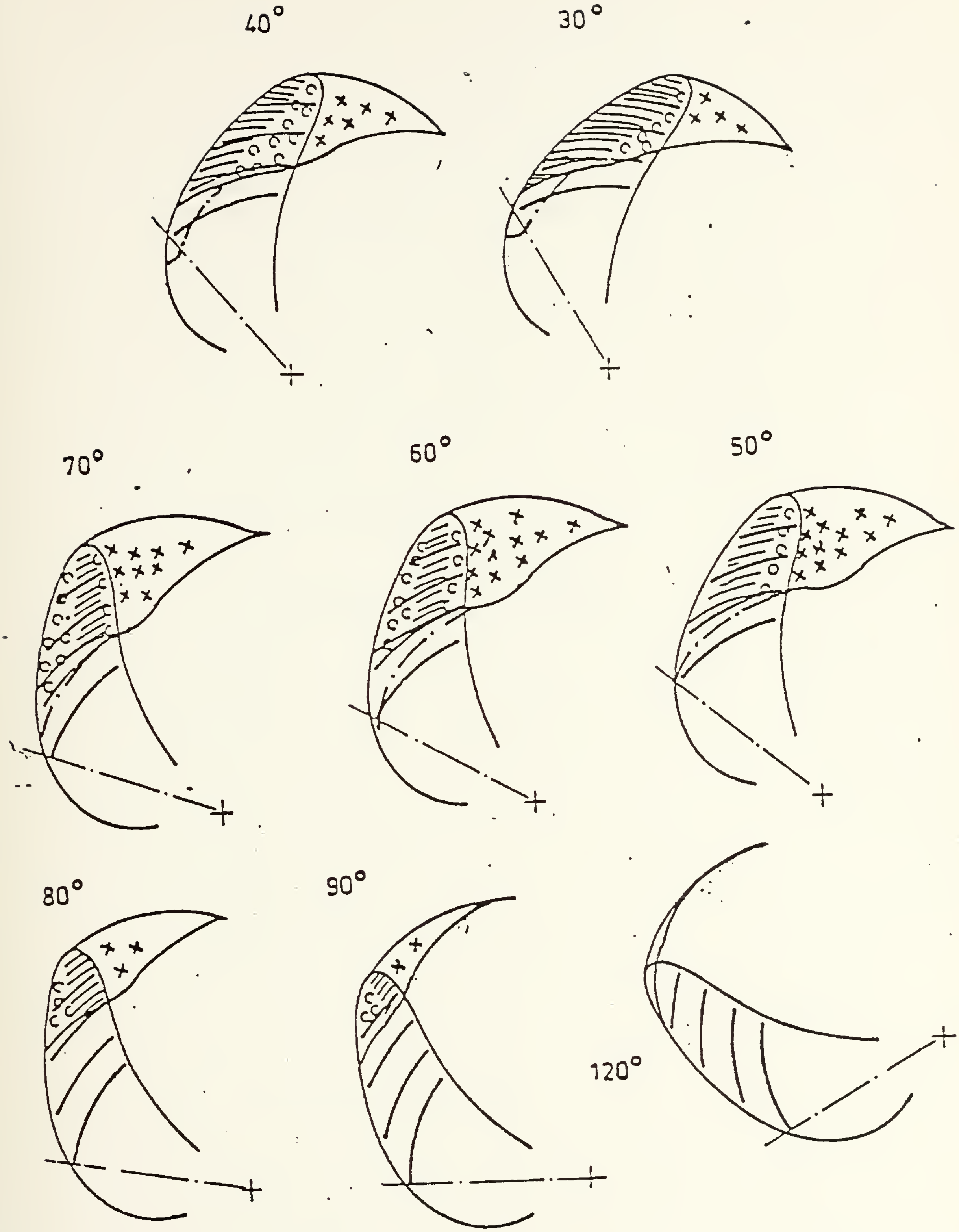
NOMINAL VS. EFFECTIVE WAKE

** APPENDIX VII **

SSPA Experimental Propeller Cavitation Extent Plots



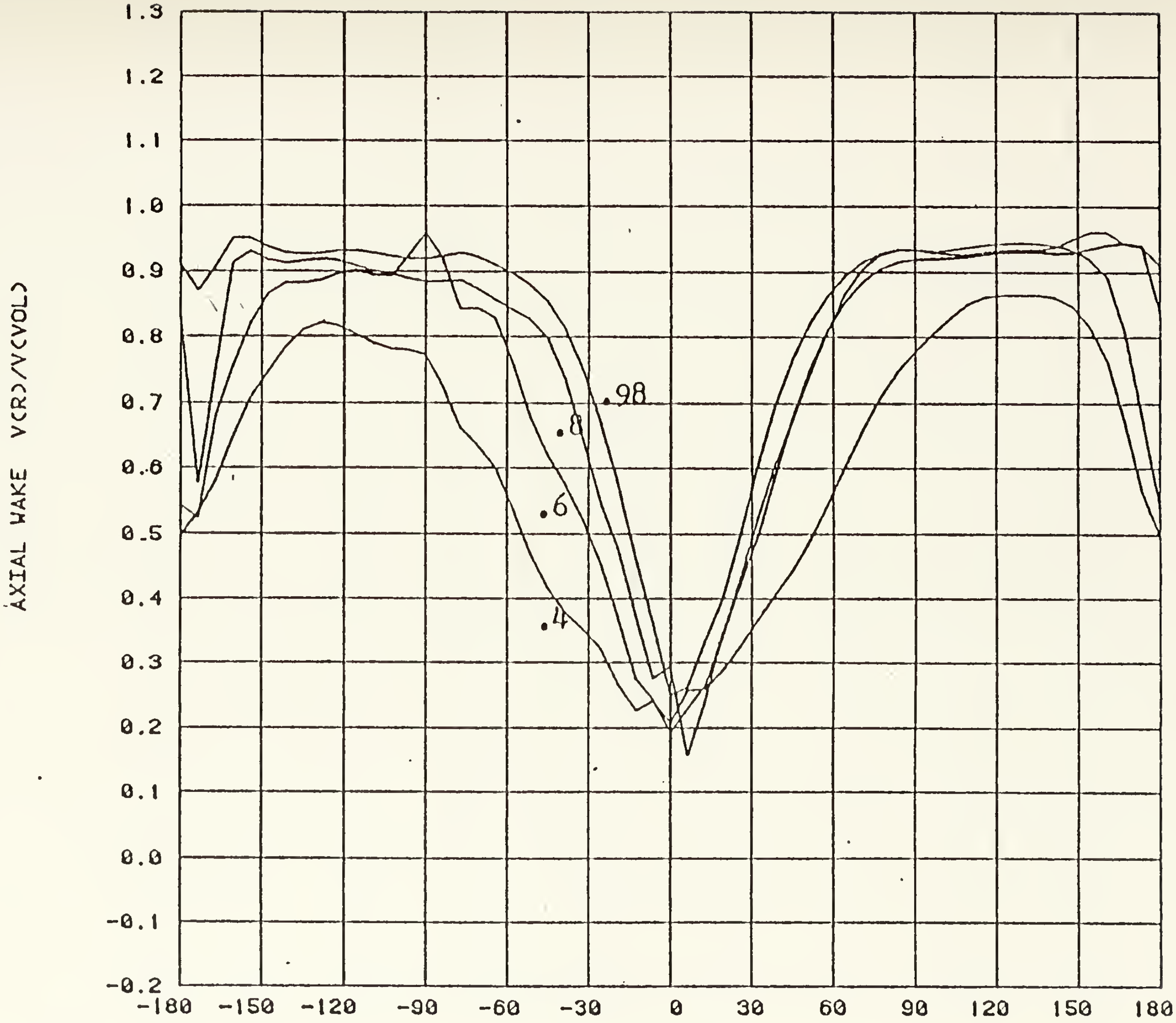
Reference (10)



Reference (10)

** APPENDIX VIII **

Nominal Wake Diagram

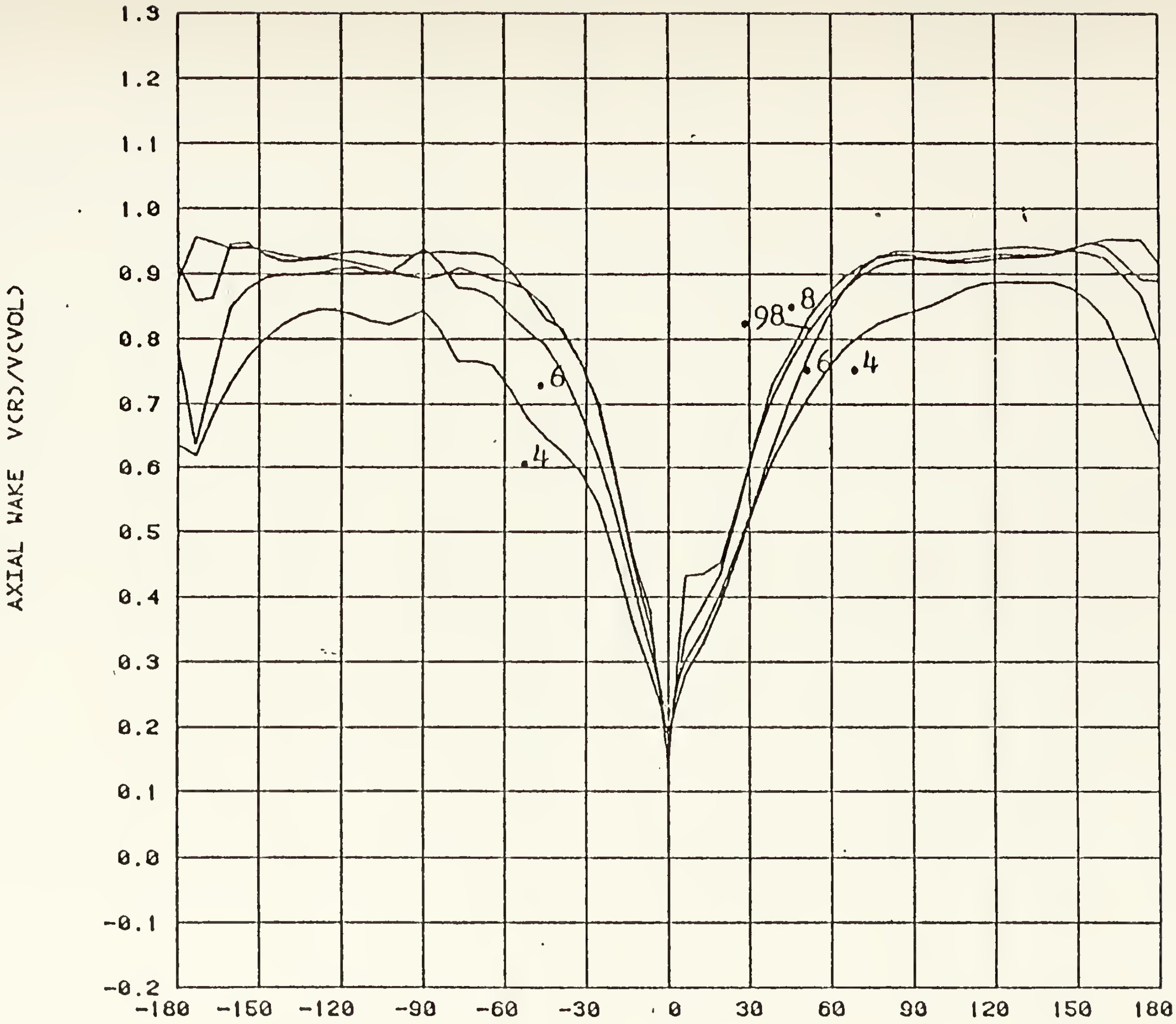


Actual Velocities (I.e., V(VOL)=1.0)

NOMINAL WAKE, .4, .6, .8, .98 RADII

** APPENDIX IX

Effective Wake Diagram



Actual Velocities (I.e., V(VOL)=1.0)

EFFECTIVE WAKE, .4, .6, .8, .98 RADII

** APPENDIX X **

UFPV Program Listing

```
C     ****************************************************
C     *                                                  *
C     *   MAIN PROGRAM IN FPUV, A PROGRAM TO CALCULATE   *
C     *   UNSTEADY VELOCITY HARMONICS AT FIELD POINTS    *
C     *   SPECIFIED INTERACTIVELY IN ROTERM.             *
C     *                                                  *
C     ****************************************************
C
C
C *** COMMON TO CONTAIN GEOMETRIC PARAMETERS
C
      COMMON / GEOCOM / MM,MN,NN,NW,NTORL,X(12,10),Y(12,10),Z(12,10),
     &   XTIP(10,11),YTIP(10,11),ZTIP(10,11),XH(101,7),YH(101,7),
     &   ZH(101,7),XW(21,10),YW(21,10),ZW(21,10),RH
C
C *** COMMON TO CONTAIN SINGULARITY STRENGTHS
C
      COMMON / SINCOM / GT(10,10,60),GB(10,9,60),GTW(10),SB(10,9),GTV
     & ,GAM(9,60),GMEAN(9),DKFW(20),HBWX,HBWY,HBWZ
C
C *** COMMON TO CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, AND VSR
C
      COMMON / FPCOM / NBLADE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM
C
C *** COMMON TO CONTAIN VELOCITY DATA
C
      COMMON / VELCOM / VIX(60),VIT(60),VIR(60),U(60,3),NSAMP
C
C *** COMMON TO CONTAIN HARMONICS COEFFICIENTS
C
      COMMON / HARCOM / A(15,3),B(15,3),AMP(15,3),PH(15,3),NH
C
C *** ISTOP IS A FLAG SET INTERACTIVELY IN ROTERM TO INDICATE END OF PRO
C
      ISTOP=0
C
C *** RDFILE WILL READ DISK DATA FILE CREATED BY PUF-2.  NOTE THAT NSR I
C     PUF-2 INPUT MUST BE DIVISIBLE BY THE NUMBER OF BLADES, NBLADE.
C
      CALL RDFILE
C
C *** INTERACTIVELY READ PARAMETERS FOR TERMINAL SESSION
C
 905  FORMAT(/' ENTER NBLADE, NTORL, IPLOT.'/
     &  ' FOR HELP, ENTER ZERO FOR NBLADE.')
  30  WRITE(6,905)
      READ(5,*) NBLADE,NTORL,IPLOT
      IF (NBLADE.LT.0) GOTO 20
      IF (NBLADE.EQ.0) GOTO 90
      IF (IPLOT.EQ.2) CALL PIEDAT(ISTOP)
      IF (ISTOP.EQ.1) GOTO 20
C
```

```
C *** ROTERM GETS DATA INTERACTIVELY FROM TERMINAL AND COMPUTES
C     NECESSARY FIELD POINT GEOMETRY.
C
   10 CALL ROTERM(ISTOP)
C
C *** IF OPERATOR INDICATES END OF PROGRAM, ISTOP WILL BE SET TO 1
C
      IF (ISTOP.GT.0) GOTO 20
C
C *** RUN THROUGH FPSTEP FOR EACH "TIME STEP."
C
      DO 100 K=1,NSR
C
C *** FPSTEP CALCULATED VELOCITY INDUCED AT FIELD POINT BY ONE BLADE
C     AT "TIME STEP" K
C
  100 CALL FPSTEP(K)
C
C *** SUMVEL WILL SUM FIELD POINT VELOCITIES INDUCED BY EACH BLADE.
C
      CALL SUMVEL
C
C *** FDUCAL WILL GENERATE HARMONIC COEFFICIENTS FOR INDUCED FIELD POINT
C     VELOCITY AS PROPELLER ROTATES.
C
      CALL FDUCAL
C
C *** PLOTVL PLOTS INDUCED FIELD POINT VELOCITIES AS A FUNCTION OF BLADE
C     ROTATION.
C
  901 FORMAT(' AXIAL COORDINATE',4X,F5.3/' RADIAL COORDINATE',2X,F6.3/
     &  ' ANGULAR COORDINATE',2X,F5.1///' AXIAL HARMONIC',6X,F8.5/
     &  ' RADIAL HARMONIC',5X,F8.5/' TANGENTIAL HARMONIC ',F8.5)
      WRITE(6,901) XFP,RFP,TZFP,(AMP(1,J),J=1,3)
      IF (IPLOT.EQ.1) CALL PLOTVL(NBLADE,NSR,NH,TFP,TZFP,U,A,B,AMP,PH,
     &  XFP,RFP,IDENT)
      GOTO 10
  906 FORMAT(/' NBLADE < 0 => STOP'
     &        /8X,     '= 0 => HELP'/8X,'> 0 => NUMBER OF BLADES'/
     &  ' NTORL  = 0 => TOTAL EFFECT'/8X,'= 1 => THICKNESS ONLY'/
     &  8X,'= 2 => LOADING ONLY'/' IPLOT  = 0 => DATA TO TERMINAL'/
     &  8X,'= 1 => PLOT -- NOT IMPLEMENTED 05/02/82 -- RLJ'
     &  /8X,'= 2 => CREATE PIEWAKE DATA FILE'/)
   90 WRITE(6,906)
      GOTO 30
   20 STOP
      END
```

```
      SUBROUTINE FPVEL(K,XF,YF,ZF,VIX,VIY,VIZ)                          UFP01010
C                                                                       UFP01020
C *** COMMON TO CONTAIN GEOMETRIC PARAMETERS                            UFP01030
C                                                                       UFP01040
      COMMON / GEOCOM / MM,MN,NN,NW,NTORL,X(12,10),Y(12,10),Z(12,10),  UFP01050
     &   XTIP(10,11),YTIP(10,11),ZTIP(10,11),XH(101,7),YH(101,7),       UFP01060
     &   ZH(101,7),XW(21,10),YW(21,10),ZW(21,10),RH                     UFP01070
C                                                                       UFP01080
C *** COMMON TO CONTAIN SINGULARITY STRENGTHS                           UFP01090
C                                                                       UFP01100
      COMMON / SINCOM / GT(10,10,60),GB(10,9,60),GTW(10),SB(10,9),GTV  UFP01110
     & ,GAM(9,60),GMEAN(9),DKFW(20),HBWX,HBWY,HBWZ                      UFP01120
C                                                                       UFP01130
C *** COMMON TO CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, AND VSR  UFP01140
C                                                                       UFP01150
      COMMON / FPCOM / NBLADE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,               UFP01160
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM                   UFP01170
C                                                                       UFP01180
      DIMENSION UTX(20,2),UTY(20,2),UTZ(20,2),UBX(20),UBY(20),UBZ(20)  UFP01190
     & ,VOFWX(20,9),VOFWY(20,9),VOFWZ(20,9)                             UFP01200
C                                                                       UFP01210
C *** INITIALIZE VARIABLES                                              UFP01220
C                                                                       UFP01230
      MID=MN/2                                                          UFP01240
      NVFW=NW                                                           UFP01250
      JMPFW=1                                                           UFP01260
      WAKX=0.                                                           UFP01270
      WAKY=0.                                                           UFP01280
      WAKZ=0.                                                           UFP01290
      VIX=0.0                                                           UFP01300
      VIY=0.0                                                           UFP01310
      VIZ=0.0                                                           UFP01320
      DO 1 M=1,MM                                                       UFP01330
      DO 1 N=1,NN                                                       UFP01340
C-----CHORDWISE VORTICES ON BLADE---------------------------------------UFP01350
      IF(NTORL.EQ.1) GO TO 8                                            UFP01360
      CALL VORSEG(XF,YF,ZF,X(N,M),Y(N,M),Z(N,M),X(N+1,M),Y(N+1,M),      UFP01370
     1 Z(N+1,M),CVX,CVY,CVZ,SX,SY,SZ,0)                                 UFP01380
C-----SPANWISE VORTICES AND SOURCES ON BLADE----------------------------UFP01390
 8    CALL VORSEG(XF,YF,ZF,X(N,M),Y(N,M),Z(N,M),X(N,M+1),Y(N,M+1),      UFP01400
     1 Z(N,M+1),SVX,SVY,SVZ,SX,SY,SZ,1)                                 UFP01410
      SBNM=SB(N,M)                                                      UFP01420
      GBNM=GB(N,M,K)                                                    UFP01430
      GTNM=GT(N,M,K)                                                    UFP01440
      IF(NTORL.EQ.0) GO TO 9                                            UFP01450
      IF(NTORL.EQ.1) GBNM=0.0                                           UFP01460
      IF(NTORL.EQ.1) GTNM=0.0                                           UFP01470
      IF(NTORL.EQ.2) SBNM=0.0                                           UFP01480
 9    VIX=VIX+GBNM*SVX+SBNM*SX+GTNM*CVX                                 UFP01490
      VIY=VIY+GBNM*SVY+SBNM*SY+GTNM*CVY                                 UFP01500
      VIZ=VIZ+GBNM*SVZ+SBNM*SZ+GTNM*CVZ                                 UFP01510
 1    CONTINUE                                                          UFP01520
C-----SEPARATED TIP VORTEX ON BLADE------------------------------        UFP01530
```

```
      IF(NTORL.EQ.1) GO TO 10                                           UFPO1540
      DO 2 N=1,NN                                                       UFPO1550
      CVX=0.0                                                           UFPO1560
      CVY=0.0                                                           UFPO1570
      CVZ=0.0                                                           UFPO1580
      NTEM=NN-N+1                                                       UFPO1590
      DO 3 LL=1,NTEM                                                    UFPO1600
      L1=LL+1                                                           UFPO1610
      CALL VORSEG(XF,YF,ZF,XTIP(N,LL),YTIP(N,LL),ZTIP(N,LL)             UFPO1620
     *  ,XTIP(N,L1),YTIP(N,L1),ZTIP(N,L1),VX,VY,VZ,SX,SY,SZ,0)         UFPO1630
      CVX=CVX+VX                                                        UFPO1640
      CVY=CVY+VY                                                        UFPO1650
 3    CVZ=CVZ+VZ                                                        UFPO1660
      VIX=VIX+GB(N,MM,K)*CVX                                            UFPO1670
      VIY=VIY+GB(N,MM,K)*CVY                                            UFPO1680
 2    VIZ=VIZ+GB(N,MM,K)*CVZ                                            UFPO1690
C-----TRANSITION WAKE---------------------------------------------------UFPO1700
      WAKX=0.                                                           UFPO1710
      WAKY=0.                                                           UFPO1720
      WAKZ=0.                                                           UFPO1730
      DO 35 N=1,100                                                     UFPO1740
      CALL VORSEG(XF,YF,ZF,XH(N,1),YH(N,1),ZH(N,1),XH(N+1,1),YH(N+1,1), UFPO1750
     1  ZH(N+1,1),VX,VY,VZ,SX,SY,SZ,0)                                  UFPO1760
      WAKX=WAKX+VX                                                      UFPO1770
      WAKY=WAKY+VY                                                      UFPO1780
   35 WAKZ=WAKZ+VZ                                                      UFPO1790
      CALL VORSEG(XF,YF,ZF,XH(1,1),0.,0.,XH(101,1),0.,0.,               UFPO1800
     &  HBWX,HBWY,HBWZ,SX,SY,SZ,0)                                      UFPO1810
      HBWX=-HBWX                                                        UFPO1820
      HBWY=-HBWY                                                        UFPO1830
      HBWZ=-HBWZ                                                        UFPO1840
   11 DO 71 M=1,MM                                                      UFPO1850
      IF(M.EQ.1) GO TO 72                                               UFPO1860
      DO 73 N=1,NW                                                      UFPO1870
      UTX(N,1)=UTX(N,2)                                                 UFPO1880
      UTY(N,1)=UTY(N,2)                                                 UFPO1890
   73 UTZ(N,1)=UTZ(N,2)                                                 UFPO1900
   72 DO 74 N=1,NW                                                      UFPO1910
      CALL VORSEG(XF,YF,ZF,XW(N,M),YW(N,M),ZW(N,M),XW(N,M+1),YW(N,M+1), UFPO1920
     1  ZW(N,M+1),UBX(N),UBY(N),UBZ(N),SX,SY,SZ,0)                      UFPO1930
      IF(M.GT.1) GO TO 75                                               UFPO1940
      CALL VORSEG(XF,YF,ZF,XW(N,M),YW(N,M),ZW(N,M),XW(N+1,M),YW(N+1,M), UFPO1950
     1  ZW(N+1,M),UTX(N,1),UTY(N,1),UTZ(N,1),SX,SY,SZ,0)                UFPO1960
   75 CALL VORSEG(XF,YF,ZF,XW(N,M+1),YW(N,M+1),ZW(N,M+1),XW(N+1,M+1),   UFPO1970
     1  YW(N+1,M+1),ZW(N+1,M+1),UTX(N,2),UTY(N,2),UTZ(N,2),SX,SY,SZ,0)  UFPO1980
   74 CONTINUE                                                          UFPO1990
      DO 76 N=1,NW                                                      UFPO2000
C                                                                       UFPO2010
C *** COMPUTE INDUCED VELOCITY FROM ONE FORCESHOE AT THIS RADIUS        UFPO2020
C                                                                       UFPO2030
      BUG=UTX(N,2)-UTX(N,1)+UBX(N)                                      UFPO2040
      CAT=UTY(N,2)-UTY(N,1)+UBY(N)                                      UFPO2050
      DOG=UTZ(N,2)-UTZ(N,1)+UBZ(N)                                      UFPO2060
```

```
      IF(N.NE.NW) GO TO 77
      IF(M.NE.MID) GO TO 78
C
C *** ALL TRANSITION WAKE VORTICES DISAPPEAR EXCEPT MID, WHICH TURNS
C     INTO ULTIMATE WAKE VORTICES (TIP AND HUB) BY ASSUMING MEAN LOADING
C
      BUG=BUG+WAKX+HBWX
      CAT=CAT+WAKY+HBWY
      DOG=DOG+WAKZ+HBWZ
      GO TO 78
   77 BUG=BUG-UBX(N+1)
      CAT=CAT-UBY(N+1)
      DOG=DOG-UBZ(N+1)
   78 VOFWX(N,M)=BUG
      VOFWY(N,M)=CAT
      VOFWZ(N,M)=DOG
   76 CONTINUE
   92 DO 93 LN=1,NVFW
      KKM=K-LN
      IF (KKM.LE.0) KKM=KKM+NSR
      VIX=VIX+VOFWX(LN,M)*(GMEAN(M)
     1                   +DKFW(LN)*(GAM(M,KKM)-GMEAN(M)))
      VIY=VIY+VOFWY(LN,M)*(GMEAN(M)
     1                   +DKFW(LN)*(GAM(M,KKM)-GMEAN(M)))
 93   VIZ=VIZ+VOFWZ(LN,M)*(GMEAN(M)
     1                   +DKFW(LN)*(GAM(M,KKM)-GMEAN(M)))
   71 CONTINUE
 10   RETURN
      END
```

```
      SUBROUTINE RDFILE                                                 UFPO2360
C     ***********************************************************      UFPO2370
C     *                                                         *      UFPO2380
C     *   SUBROUTINE RDFILE READS DISK FILE14 DATA CREATED      *      UFPO2390
C     *   BY PUF-2 AND XAVES DATA NEEDED BY FPUV IN             *      UFPO2400
C     *   COMMONS GEDCOM AND SINCOM.  NOTE THAT NSR IN          *      UFPO2410
C     *   PUF-2 MUST BE DIVISIBLE BY THE NUMBER OF BLADES,      *      UFPO2420
C     *   NBLADE.                                               *      UFPO2430
C     *                                                         *      UFPO2440
C     *                                                         *      UFPO2450
C     *   NN         =>                                         *      UFPO2460
C     *   MM         =>                                         *      UFPO2470
C     *   NW         => NUMBER OF VORTICES IN KEY WAKE          *      UFPO2480
C     *   NBLADE     => NUMBER OF PROPELLER BLADES              *      UFPO2490
C     *   NVC        => NO OF CHORD VORTICES, OTHER BLADES      *      UFPO2500
C     *   NVS        => NO OF SPAN VORTICES, OTHER BLADES       *      UFPO2510
C     *   NVW        => NO OF WAKE VORTICES, OTHER WAKE         *      UFPO2520
C     *   NSR        => NUMBER OF TIME STEPS/REVOLUTION         *      UFPO2530
C     *   NWK        => NUMBER OF RADII IN WAKE HARMONICS       *      UFPO2540
C     *                                                         *      UFPO2550
C     ***********************************************************      UFPO2560
C                                                                       UFPO2570
C                                                                       UFPO2580
C *** COMMON TO CONTAIN GEOMETRIC PARAMETERS                            UFPO2590
C                                                                       UFPO2600
      COMMON / GEOCOM / MM,MN,NN,NW,NTDRL,X(12,10),Y(12,10),Z(12,10),   UFPO2610
     &   XTIP(10,11),YTIP(10,11),ZTIP(10,11),XH(101,7),YH(101,7),       UFPO2620
     &   ZH(101,7),XW(21,10),YW(21,10),ZW(21,10),RH                     UFPO2630
C                                                                       UFPO2640
C *** COMMON TO CONTAIN SINGULARITY STRENGTHS                           UFPO2650
C                                                                       UFPO2660
      COMMON / SINCOM / GT(10,10,60),GB(10,9,60),GTW(10),SB(10,9),GTV   UFPO2670
     &  ,GAM(9,60),GMEAN(9),DKFW(20),HBWX,HBWY,HBWZ                     UFPO2680
C                                                                       UFPO2690
C *** COMMON TO CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, AND VSR   UFPO2700
C                                                                       UFPO2710
      COMMON / FPCOM / NBLADE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,               UFPO2720
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM                   UFPO2730
C                                                                       UFPO2740
C                                                                       UFPO2750
      DIMENSION RZ(10),R(9),CHDCP(9),CHORZ(10),PHIB(90)                 UFPO2760
      DIMENSION XD(6,4,6),YO(6,4,6),ZD(6,4,6)                           UFPO2770
      DIMENSION RWK(11),AWA(16,11),BWA(16,11),AWR(16,11),BWR(16,11),    UFPO2780
     &  AWT(16,11),BWT(16,11)                                           UFPO2790
      DIMENSION XWO(5,4,6),YWD(5,4,6),ZWO(5,4,6)                        UFPO2800
      DIMENSION SBO(5,3),GBC(5,3,60),GAMWC(4,3,60),GTC(5,4,60),         UFPO2810
     &  GAMC(3,60),GMEANO(3),DKOW(4)                                    UFPO2820
      READ(14) AJ                                                       UFPO2830
      IF (AJ.GE.2.0) STOP                                               UFPO2840
      READ(14) ISTDY,NBLADE,MM,NN,NW,NVC,NVS,NVW,NSR,NWK                UFPO2850
C                                                                       UFPO2860
C *** SET CONSTANT FOR GRID                                             UFPO2870
C                                                                       UFPO2880
```

```
      NM=NN-1                                                           UFPO2890
      MN=MM+1                                                           UFPO2900
      NP=NN+1                                                           UFPO2910
      NZ=NW+1                                                           UFPO2920
      NVCP=NVC+1                                                        UFPO2930
      NVSP=NVS+1                                                        UFPO2940
      NBLO=NBLADE-1                                                     UFPO2950
C                                                                       UFPO2960
C *** SET CONSTANTS FOR UNSTEADY PROBLEM                                UFPO2970
C                                                                       UFPO2980
      IF (ISTDY.EQ.1) NSR=1                                             UFPO2990
      NVWP=NVW+1                                                        UFPO3000
      NTB=NN*MM                                                         UFPO3010
C                                                                       UFPO3020
C *** READ KEY BLADE GEOMETRY                                           UFPO3030
C                                                                       UFPO3040
      READ(14) ((X(N,M),N=1,NP),M=1,MN),                                UFPO3050
     2         ((Y(N,M),N=1,NP),M=1,MN),                                UFPO3060
     3         ((Z(N,M),N=1,NP),M=1,MN),                                UFPO3070
     4         (RZ(M),M=1,MN),                                          UFPO3080
     5         (R(M),M=1,MM),                                           UFPO3090
     6         (CHOCP(M),M=1,MM),                                       UFPO3100
     7         (CHORZ(M),M=1,MN),                                       UFPO3110
     8         (PHIB(K),K=1,NTB)                                        UFPO3120
C                                                                       UFPO3130
C *** READ BLADE GEDMETRY OF DTHER BLADES                               UFPO3140
C                                                                       UFPO3150
      IF (NBLADE.LE.1) GOTO 11                                          UFPO3160
      READ(14) (((XD(N,M,K),N=1,NVCP),M=1,NVSP),K=1,NBLD),              UFPO3170
     2         (((YO(N,M,K),N=1,NVCP),M=1,NVSP),K=1,NBLO),              UFPO3180
     3         (((ZO(N,M,K),N=1,NVCP),M=1,NVSP),K=1,NBLD)               UFPO3190
   11 CONTINUE                                                          UFPO3200
C                                                                       UFPO3210
C *** READ WAKE RADII AND WAKE HARMONICS                                UFPO3220
C                                                                       UFPO3230
      READ(14) (RWK(M),M=1,NWK)                                         UFPO3240
      READ(14) ((AWA(I,M),I=1,16),M=1,NWK),                             UFPO3250
     2         ((BWA(I,M),I=1,16),M=1,NWK),                             UFPO3260
     3         ((AWR(I,M),I=1,16),M=1,NWK),                             UFPO3270
     4         ((BWR(I,M),I=1,16),M=1,NWK),                             UFPO3280
     5         ((AWT(I,M),I=1,16),M=1,NWK),                             UFPO3290
     6         ((BWT(I,M),I=1,16),M=1,NWK)                              UFPO3300
C                                                                       UFPO3310
C *** READ VARIOUS PROPELLER/WAKE CONSTANTS                             UFPO3320
C                                                                       UFPO3330
      READ(14) WAKE,RPM,DIAM,RH,CDRAG,VSR,UR,(IDENT(N),N=1,18)          UFPO3340
C                                                                       UFPO3350
C *** READ KEY WAKE GEOMETRY                                            UFPO3360
C                                                                       UFPO3370
      READ(14) ((XW(N,M),N=1,NZ),M=1,MN),                               UFPO3380
     2         ((YW(N,M),N=1,NZ),M=1,MN),                               UFPO3390
     3         ((ZW(N,M),N=1,NZ),M=1,MN),                               UFPO3400
     4         ((XH(N,K),N=1,101),K=1,NBLADE),                          UFPO3410
```

```
     5          ((YH(N,K),N=1,101),K=1,NBLADE),                         UFP03420
     6          ((ZH(N,K),N=1,101),K=1,NBLADE),                         UFP03430
     7          ((XTIP(N,L),L=1,NP),N=1,NN),                            UFP03440
     8          ((YTIP(N,L),L=1,NP),N=1,NN),                            UFP03450
     9          ((ZTIP(N,L),L=1,NP),N=1,NN),                            UFP03460
C                                                                       UFP03470
C *** READ KEY BLADE SINGULARITY STRENGTHS                              UFP03480
C                                                                       UFP03490
     *          ((SB(N,M),N=1,NN),M=1,MM),                              UFP03500
     1          (((GB(N,M,KK),N=1,NN),M=1,MM),KK=1,NSR),                UFP03510
     2          (((GT(N,M,KK),N=1,NN),M=1,MN),KK=1,NSR),                UFP03520
     3          ((GAM(M,KK),M=1,MM),KK=1,NSR)                           UFP03530
C                                                                       UFP03540
C *** OTHER WAKE GEOMETRY                                               UFP03550
C                                                                       UFP03560
      IF (NBLADE.LE.1) GOTO 12                                          UFP03570
      READ(14) (((XWO(N,M,K),N=1,NVWP),M=1,NVSP),K=1,NBLO),             UFP03580
     2          (((YWO(N,M,K),N=1,NVWP),M=1,NVSP),K=1,NBLO),            UFP03590
     3          (((ZWO(N,M,K),N=1,NVWP),M=1,NVSP),K=1,NBLO),            UFP03600
C                                                                       UFP03610
C *** OTHER BLADE SINGULARITIES                                         UFP03620
C                                                                       UFP03630
     4          ((SBO(N,M),N=1,NVC),M=1,NVS),                           UFP03640
     5          (((GBC(N,M,KK),N=1,NVC),M=1,NVS),KK=1,NSR),             UFP03650
     6          (((GAMWC(N,M,KK),N=1,NVW),M=1,NVS),KK=1,NSR),           UFP03660
     7          (((GTC(N,M,KK),N=1,NVC),M=1,NVSP),KK=1,NSR),            UFP03670
     8          ((GAMC(M,KK),M=1,NVS),KK=1,NSR),                        UFP03680
     9          (GMEAN(M),M=1,MM),                                      UFP03690
     *          (GMEANO(M),M=1,NVS),                                    UFP03700
     1          (DKFW(N),N=1,NW),                                       UFP03710
     2          (DKOW(N),N=1,NVW)                                       UFP03720
   12 CONTINUE                                                          UFP03730
   99 GOTO 100                                                          UFP03740
C                                                                       UFP03750
C &&& DIAGNOSTICS &&&&&&&&&&&&&&&&&&&&&&&&                              UFP03760
C                                                                       UFP03770
  901 FORMAT(16I5)                                                      UFP03780
  902 FORMAT(8F10.5)                                                    UFP03790
  903 FORMAT(18A4)                                                      UFP03800
      WRITE(1,902) AJ                                                   UFP03810
      WRITE(1,901) ISTDY,NBLADE,MM,NN,NW,NVC,NVS,NVW,NSR,NWK            UFP03820
      WRITE(1,902) ((X(N,M),N=1,NP),M=1,MN),                            UFP03830
     2          ((Y(N,M),N=1,NP),M=1,MN),                               UFP03840
     3          ((Z(N,M),N=1,NP),M=1,MN),                               UFP03850
     4          (RZ(M),M=1,MN),                                         UFP03860
     5          (R(M),M=1,MM),                                          UFP03870
     6          (CHOCP(M),M=1,MM),                                      UFP03880
     7          (CHORZ(M),M=1,MN),                                      UFP03890
     8          (PHIB(K),K=1,NTB)                                       UFP03900
      IF (NBLADE.LE.1) GOTO 21                                          UFP03910
      WRITE(1,902) (((XO(N,M,K),N=1,NVCP),M=1,NVSP),K=1,NBLO),          UFP03920
     2          (((YO(N,M,K),N=1,NVCP),M=1,NVSP),K=1,NBLO),             UFP03930
     3          (((ZO(N,M,K),N=1,NVCP),M=1,NVSP),K=1,NBLO)              UFP03940
```

```
 21  CDNTINUE                                                         UFPO3950
     WRITE(1,902) (RWK(M),M=1,NWK)                                    UFPO3960
     WRITE(1,902) ((AWA(I,M),I=1,16),M=1,NWK),                        UFPO3970
    3          ((BWA(I,M),I=1,16),M=1,NWK),                           UFPO3980
    4          ((AWR(I,M),I=1,16),M=1,NWK),                           UFPO3990
    5          ((BWR(I,M),I=1,16),M=1,NWK),                           UFPO4000
    6          ((AWT(I,M),I=1,16),M=1,NWK),                           UFPO4010
    7          ((BWT(I,M),I=1,16),M=1,NWK)                            UFPO4020
     WRITE(1,902) WAKE,RPM,DIAM,RH,CDRAG,VSR,UR                       UFPO4030
     WRITE(1,903) (IDENT(N),N=1,18)                                   UFPO4040
     WRITE(1,902) ((XW(N,M),N=1,NZ),M=1,MN),                          UFPO4050
    2          ((YW(N,M),N=1,NZ),M=1,MN),                             UFPO4060
    3          ((ZW(N,M),N=1,NZ),M=1,MN),                             UFPO4070
    4          ((XH(N,K),N=1,101),K=1,NBLADE),                        UFPO4080
    5          ((YH(N,K),N=1,101),K=1,NBLADE),                        UFPO4090
    6          ((ZH(N,K),N=1,101),K=1,NBLADE),                        UFPO4100
    7          ((XTIP(N,L),L=1,NP),N=1,NN),                           UFPO4110
    8          ((YTIP(N,L),L=1,NP),N=1,NN),                           UFPO4120
    9          ((ZTIP(N,L),L=1,NP),N=1,NN)                            UFPO4130
     WRITE(1,902) ((SB(N,M),N=1,NN),M=1,MM),                          UFPO4140
    2          (((GB(N,M,KK),N=1,NN),M=1,MM),KK=1,NSR),               UFPO4150
    3          (((GT(N,M,KK),N=1,NN),M=1,MN),KK=1,NSR),               UFPO4160
    4          ((GAM(M,KK),M=1,MM),KK=1,NSR)                          UFPO4170
     IF (NBLADE.LE.1) GDTD 22                                         UFPO4180
     WRITE(1,902) (((XWO(N,M,K),N=1,NVWP),M=1,NVSP),K=1,NBLO),        UFPO4190
    2          (((YWD(N,M,K),N=1,NVWP),M=1,NVSP),K=1,NBLD),           UFPO4200
    3          (((ZWO(N,M,K),N=1,NVWP),M=1,NVSP),K=1,NBLO)            UFPO4210
     WRITE(1,902) ((SBD(N,M),N=1,NVC),M=1,NVS),                       UFPO4220
    2          (((GBC(N,M,KK),N=1,NVC),M=1,NVS),KK=1,NSR),            UFPO4230
    3          (((GAMWC(N,M,KK),N=1,NVW),M=1,NVS),KK=1,NSR),          UFPO4240
    4          (((GTC(N,M,KK),N=1,NVC),M=1,NVSP),KK=1,NSR),           UFPO4250
    5          ((GAMC(M,KK),M=1,NVS),KK=1,NSR),                       UFPO4260
    6          (GMEAN(M),M=1,MM),                                     UFPO4270
    7          (GMEANO(M),M=1,NVS),                                   UFPO4280
    8          (DKFW(N),N=1,NW),                                      UFPO4290
    9          (DKOW(N),N=1,NVW)                                      UFPO4300
 22  CONTINUE                                                         UFPO4310
100  RETURN                                                           UFPO4320
     END                                                              UFPO4330
```

```
      SUBROUTINE ROTERM(ISTOP)
C     ******************************************************
C     *                                                    *
C     *   SUBROUTINE ROTERM READS IN BLADE AND GEOMETRY    *
C     *   DATA INTERACTIVELY AND COMPUTES NECESSARY FIELD  *
C     *   POINT GEOMETRY.                                  *
C     *                                                    *
C     *   NBLADE      => NUMBER OF BLADES                  *
C     *   NTORL   = 0 => CONSIDER THICKNESS AND LOADING    *
C     *           = 1 => CONSIDER THICKNESS ONLY           *
C     *           = 2 => CONSIDER LOADING ONLY             *
C     *   XFP         => AXIAL FIELD POINT COORDINATE,     *
C     *                  POSITIVE DOWNSTREAM               *
C     *   RFP         => RADIAL FIELD POINT COORDINATE     *
C     *   TZFP        => INITAL FIELD POINT ANGLE IN DEG.  *
C     *   YBAR,ZBAR   => Y AND Z FIELD POINT COORDINATES   *
C     *   TH          => ANGLE FROM TIME STEP INPUT        *
C     *   YB,ZB       => Y,Z FIELD PT COORD, BEFORE TIME ST *
C     *                                                    *
C     ******************************************************
C
C *** COMMON TO CONTAIN GEOMETRIC PARAMETERS
C
      COMMON / GEOCOM / MM,MN,NN,NW,NTORL,X(12,10),Y(12,10),Z(12,10),
     &   XTIP(10,11),YTIP(10,11),ZTIP(10,11),XH(101,7),YH(101,7),
     &   ZH(101,7),XW(21,10),YW(21,10),ZW(21,10),RH
C
C *** COMMON TO CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, AND VSR
C
      COMMON / FPCOM / NBLADE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM
C
C
 901  FORMAT(/' ENTER XFP, RFP, TZFP.'/
     &   ' FOR HELP, ENTER -11. FOR XFP.')
  10  WRITE(6,901)
      READ(5,*) XFP,RFP,TZFP
      IF (XFP.LT.-10.) GOTO 90
      IF (TZFP.LT.0.0) GOTO 70
      IF (RFP.GT.0.) GOTO 60
  20  WRITE(6,902)
 902  FORMAT(/' ENTER NBW,N,M,KK'/' FOR HELP, ENTER ZERO FOR NBW.')
      READ(5,*) NBW,N,M,KK
      TH=6.283185*FLOAT(KK-1)/FLOAT(NSR)
      IF (NBW.EQ.0) GOTO 80
C
C *** FIELD POINT CENTERED ON BLADE GRID.
C
      IF(NBW.EQ.1) CALL FPGRID(X,Y,Z,N,M,KK,12)
C
C *** FIELD POINT CENTERED ON TRANSITION WAKE GRID.
C
      IF(NBW.EQ.2) CALL FPGRID(XW,YW,ZW,N,M,KK,21)
```

```
      IF(NBW.NE.1.AND.NBW.NE.2) GOTO 10
   60 DTFP=360.0/FLOAT(NSR)
      RETURN
   70 ISTOP=1
      RETURN
   80 WRITE(6,903)
  903 FORMAT(/' NBW = 0 => HELP'/5X,'= 1 => BLADE WAKE GRID'/5X,
     & '= 2 => TRANSITION WAKE GRID'/' N',7X,'=> CHORDWISE COORDINATE'/
     & ' M',7X,'=> SPANWISE COORDINATE'/' KK',6X,
     & '=> TIME STEPS ROTATED + 1'/)
      GOTO 20
   90 WRITE(6,904)
  904 FORMAT(/' XFP    <-10=> HELP'/8X,
     & '>-10=> AXIAL FIELD POINT COORDINATE, POSITIVE DOWNSTREAM'/
     & ' RFP    < 0 => SELECT COORDINATE USING BLADE OR WAKE GRID'/
     & 8X,'> 0 => RADIAL FIELD POINT COORDINATE'/
     & ' TZFP   < 0 => STOP'/8X,'> 0 => ANGULAR FILE POINT COORDINATE'
     & /)
      GOTO 10
      END
```

```
      SUBROUTINE FPGRID(XX,YY,ZZ,N,M,KK,NDIM)                           UFP05070
C     ***********************************************************       UFP05080
C     *                                                         *       UFP05090
C     *   SUBROUTINE GRID CALCULATES X,R, AND THETA FIELD       *       UFP05100
C     *   POINT GEOMETRY COMPONENTS GIVEN GRID VECTORS          *       UFP05110
C     *   X,Y, AND Z FOR BLADE OR WAKE, GRID COORDINATES        *       UFP05120
C     *   N AND M, AND TIME STEP KK.                            *       UFP05130
C     *                                                         *       UFP05140
C     *   XX,YY,ZZ  => GRID GEOMETRY VECTORS                    *       UFP05150
C     *   N,M       => GRID INDICES SELECTED FOR VELOCITY       *       UFP05160
C     *   KK        => TIME STEP. NUMBER OF ANGLE STEPS + 1*            UFP05170
C     *   XFP,YBAR,ZBAR,TZFP,RFP                                *       UFP05180
C     *             => X,Y,Z,THETA,RADIAL FIELD PT COORD        *       UFP05190
C     *   DBLADE    => ANGLE BETWEEN BLADES                     *       UFP05200
C     *   TH        => ANGLE OF TIME STEP                       *       UFP05210
C     *   NDIM      => FIRST DIMENSION OF GRIDS PASSED          *       UFP05220
C     *                                                         *       UFP05230
C     ***********************************************************       UFP05240
C                                                                       UFP05250
C *** COMMON TO CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, AND VSR   UFP05260
C                                                                       UFP05270
      COMMON / FPCOM / NBLADE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,               UFP05280
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM                   UFP05290
C                                                                       UFP05300
      DIMENSION XX(NDIM,10),YY(NDIM,10),ZZ(NDIM,10)                     UFP05310
C                                                                       UFP05320
C *** COMPUTE X,Y,Z COORDINATES IN ZERO ANGLE POSITION                  UFP05330
C                                                                       UFP05340
      TH=6.283185*FLOAT(KK-1)/FLOAT(NSR)                                UFP05350
      THDEG=TH*360./6.283185                                            UFP05360
      DBLADE=360.0/FLOAT(NBLADE)                                        UFP05370
      XFP=0.25*(XX(N,M) + XX(N+1,M) + XX(N,M+1) + XX(N+1,M+1))          UFP05380
      YB =0.25*(YY(N,M) + YY(N+1,M) + YY(N,M+1) + YY(N+1,M+1))          UFP05390
      ZB =0.25*(ZZ(N,M) + ZZ(N+1,M) + ZZ(N,M+1) + ZZ(N+1,M+1))          UFP05400
C                                                                       UFP05410
C *** CONVERT TO CYLINDRICAL COORDINATES                                UFP05420
C                                                                       UFP05430
      RFP=SQRT(YB*YB+ZB*ZB)                                             UFP05440
      TZFP=ATAN2(ZB,YB)/1.7453293E-02                                   UFP05450
      TZFP=TZFP+THDEG                                                   UFP05460
C                                                                       UFP05470
C *** BRANCH IF GENERATING PIEWAKE DATA                                 UFP05480
C                                                                       UFP05490
      IF (IPLOT.EQ.2) GOTO 200                                          UFP05500
      TZFP=180.0/FLOAT(NBLADE) - ATAN2(ZB,YB)/1.7453293E-02             UFP05510
C                                                                       UFP05520
C *** KEEP TZFP BETWEEN 0 AND ONE BLADE ANGLE                           UFP05530
C                                                                       UFP05540
      DO 100 K=1,NBLADE                                                 UFP05550
      IF(TZFP.LT.0.0) TZFP=TZFP+DBLADE                                  UFP05560
      IF(TZFP.GT.DBLADE) TZFP=TZFP-DBLADE                               UFP05570
  100 CONTINUE                                                          UFP05580
C                                                                       UFP05590
```

```
C *** ROTATE BLADE FOR PIEWAKE DATA                                      UFP05600
C                                                                       UFP05610
 200  YBAR=YB*COS(TH) - ZB*SIN(TH)                                       UFP05620
      ZBAR=YB*SIN(TH) + ZB*COS(TH)                                       UFP05630
 300  RETURN                                                             UFP05640
      END                                                                UFP05650
```

```
      SUBROUTINE FPSTEP(K)                                              UFP05660
C     ******************************************************           UFP05670
C     *                                                    *           UFP05680
C     *   FPSTEP IS A SUBROUTINE TO CALCULATE THE FIELD    *           UFP05690
C     *   POINT VELOCITIES INDUCED BY ONE BLADE AT "TIME   *           UFP05700
C     *   STEP", K.  THE "DIRTY WORK" IS DONE IN           *           UFP05710
C     *   SUBROUTINE FPVEL.                                *           UFP05720
C     *                                                    *           UFP05730
C     *   TFP      => ANGULAR FIELD POINT COORDINATE, DEG. *           UFP05740
C     *   TRAD     => ANGULAR FIELD POINT COORDINATE, RAD. *           UFP05750
C     *   YFP,ZFP  => Y AND Z FIELD POINT COORDINATES      *           UFP05760
C     *   VIX,VIY,VIZ,VIR,VIT => INDUCED VELOCITY COMPONENT*           UFP05770
C     *                                                    *           UFP05780
C     ******************************************************           UFP05790
C                                                                       UFP05800
C *** COMMON TO CONTAIN GEOMETRIC PARAMETERS                            UFP05810
C                                                                       UFP05820
      COMMON / GEOCOM / MM,MN,NN,NW,NTORL,X(12,10),Y(12,10),Z(12,10),   UFP05830
     &   XTIP(10,11),YTIP(10,11),ZTIP(10,11),XH(101,7),YH(101,7),       UFP05840
     &   ZH(101,7),XW(21,10),YW(21,10),ZW(21,10),RH                     UFP05850
C                                                                       UFP05860
C *** COMMON TO CONTAIN SINGULARITY STRENGTHS                           UFP05870
C                                                                       UFP05880
      COMMON / SINCOM / GT(10,10,60),GB(10,9,60),GTW(10),SB(10,9),GTV   UFP05890
     &  ,GAM(9,60),GMEAN(9),DKFW(20),HBWX,HBWY,HBWZ                     UFP05900
C                                                                       UFP05910
C *** COMMON TO CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, AND VSR   UFP05920
C                                                                       UFP05930
      COMMON / FPCOM / NBLADE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,               UFP05940
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM                   UFP05950
C                                                                       UFP05960
C *** COMMON TO CONTAIN VELOCITY DATA                                   UFP05970
C                                                                       UFP05980
      COMMON / VELCOM / VIX(60),VIT(60),VIR(60),U(60,3),NSAMP           UFP05990
C                                                                       UFP06000
C *** CALCULATE CARTESIAN FIELD POINT COORDINATES TO FEED FPVEL         UFP06010
C                                                                       UFP06020
      TFP(K)=FLOAT(K-1)*DTFP-TZFP                                       UFP06030
      TRAD=TFP(K)*1.7453293E-02                                         UFP06040
      COSFP=COS(TRAD)                                                   UFP06050
      SINFP=SIN(TRAD)                                                   UFP06060
      YFP=RFP*COSFP                                                     UFP06070
      ZFP=RFP*SINFP                                                     UFP06080
C                                                                       UFP06090
C *** FPVEL WILL CALCULATE VIX, VIY, AND VIZ FOR FIELD POINT            UFP06100
C                                                                       UFP06110
      CALL FPVEL(K,XFP,YFP,ZFP,VIX(K),VIY,VIZ)                          UFP06120
      VIR(K)=VIY*COSFP + VIZ*SINFP                                      UFP06130
      VIT(K)=VIZ*COSFP - VIY*SINFP                                      UFP06140
      RETURN                                                            UFP06150
      END                                                               UFP06160
```

```
      SUBROUTINE SUMVEL
C     *******************************************************
C     *                                                     *
C     *   SUBROUTINE SUMVEL SUMS VELOCITIES FROM EACH       *
C     *   BLADE, CALCULATED IN FPSTEP, TO PRODUCE THE       *
C     *   TOTAL PROPELLER-INDUCED FIELD POINT VELOCITY.     *
C     *                                                     *
C     *   VIX,VIR,VIT => VELOCITY COMPONENTS INDUCED BY     *
C     *                  ONE BLADE AT "TIME STEP" K         *
C     *   U(N,J)      => VELOCITY INDUCED BY ALL BLADES     *
C     *                  AT "TIME STEP" N                   *
C     *           J=0 => AXIAL COMPONENT                    *
C     *           J=1 => RADIAL COMPONENT                   *
C     *           J=2 => TANGENTIAL COMPONENT               *
C     *   NSAMP       => NUMBER OF "TIME STEPS" IN ONE      *
C     *                  BLADE ANGLE (= NUMBER OF           *
C     *                  U VELOCITIES COMPUTED.)            *
C     *                                                     *
C     *******************************************************
C
C *** COMMON TO CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, AND VSR
C
      COMMON / FPCOM / NBLADE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM
C
C *** COMMON TO CONTAIN VELOCITY DATA
C
      COMMON / VELCOM / VIX(60),VIT(60),VIR(60),U(60,3),NSAMP
C
C *** AT ANY GIVEN REAL TIME, EACH BLADE CAN BE CONSIDERED TO HAVE
C     TRAVELLED THROUGH A CERTAIN NUMBER OF "TIME STEPS".
C     BY SUMMING VIX, VIR, AND VIT AT BLADE RATE, WE CAN INCLUDE
C     THE VELOCITIES INDUCED BY EACH BLADE INTO ONE VELOCITY
C     INDUCED BY THE ENTIRE PROPELLER.
C
      NSAMP=NSR/NBLADE
      DO 20 N=1,NSAMP
      DO 10 J=1,3
   10 U(N,J)=0.0
      DO 20 K=1,NBLADE
      L=N+(K-1)*NSAMP
      U(N,1)=U(N,1)+VIX(L)
      U(N,2)=U(N,2)+VIR(L)
   20 U(N,3)=U(N,3)+VIT(L)
C
C *** NOW CONVERT FROM REFERENCE VELOCITY TO SHIP VELOCITY
C
      DO 30 N=1,NSAMP
      DO 30 J=1,3
   30 U(N,J)=U(N,J)/VSR
      RETURN
      END
```

```
      SUBROUTINE FOUCAL                                                 UFP06690
C     ***************************************************               UFP06700
C     *                                                 *               UFP06710
C     *   SUBROUTINE FOUCAL TAKES THE PROPELLER INDUCED *               UFP06720
C     *   VELOCITIES CALCULATED IN SUMVEL AND PERFORMS  *               UFP06730
C     *   A HARMONIC ANALYSIS ON THEM.  IT CALCULATES   *               UFP06740
C     *   FOURIER COEFFICIENTS FOR BOTH COSINE (A) AND  *               UFP06750
C     *   SINE (B) AND ALSO COSINE AMPLITUDE (AMP) AND  *               UFP06760
C     *   PHASE ANGLE (PH).                             *               UFP06770
C     *                                                 *               UFP06780
C     *   NH              => NUMBER OF HARMONICS CALCULATED *           UFP06790
C     *   A(N,J),B(N,J)   => FOURIER COEFFICIENTS       *               UFP06800
C     *                      A => COSINE COEFFICIENTS   *               UFP06810
C     *                      B => SINE COEFFICIENTS     *               UFP06820
C     *                      N => HARMONIC NUMBER UP TO 15 *            UFP06830
C     *                  J = 1 => AXIAL VELOCITY        *               UFP06840
C     *                    = 2 => RADIAL VELOCITY       *               UFP06850
C     *                    = 3 => TANGENTIAL VELOCITY   *               UFP06860
C     *   AMP(N,J)        => FOURIER COEFFICIENT AMPLITUDE *            UFP06870
C     *                      N AND J SUBSCRIPTS AS FOR A, B *           UFP06880
C     *   PH(N,J)         => PHASE ANGLE ASSOCIATED WITH AMP *          UFP06890
C     *   NFB             => NUMBER OF FIELD POINTS / BLADE *           UFP06900
C     *                                                 *               UFP06910
C     ***************************************************               UFP06920
C                                                                       UFP06930
C *** COMMON TO CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, AND VSR   UFP06940
C                                                                       UFP06950
      COMMON / FPCOM / NBLADE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,               UFP06960
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM                   UFP06970
C                                                                       UFP06980
C *** COMMON TO CONTAIN VELOCITY DATA                                   UFP06990
C                                                                       UFP07000
      COMMON / VELCOM / VIX(60),VIT(60),VIR(60),U(60,3),NSAMP           UFP07010
C                                                                       UFP07020
C *** COMMON TO CONTAIN HARMONICS COEFFICIENTS                          UFP07030
C                                                                       UFP07040
      COMMON / HARCOM / A(15,3),B(15,3),AMP(15,3),PH(15,3),NH           UFP07050
C                                                                       UFP07060
C *** FIRST CALCULATE A AND B                                           UFP07070
C                                                                       UFP07080
      NFB=NSR/NBLADE                                                    UFP07090
      NHMAX=NFB/2+1                                                     UFP07100
      NH=MIN0(NHMAX,15)                                                 UFP07110
      STEP=6.283185/NFB                                                 UFP07120
      ANFB=2.0/NFB                                                      UFP07130
      BNFB=1.0/NFB                                                      UFP07140
      DO 11 K=1,NH                                                      UFP07150
      DO 10 J=1,3                                                       UFP07160
      A(K,J)=0.0                                                        UFP07170
 10   B(K,J)=0.0                                                        UFP07180
      DO 9 N=1,NFB                                                      UFP07190
      T=((N-1)*(K-1))*STEP                                              UFP07200
      ST=SIN(T)                                                         UFP07210
```

```
      CT=COS(T)                                                         UFP07220
      DO 9 J=1,3                                                        UFP07230
      A(K,J)=A(K,J)+U(N,J)*CT                                           UFP07240
 9    B(K,J)=B(K,J)+U(N,J)*ST                                           UFP07250
C                                                                       UFP07260
C *** CONVERT A AND B INTO AMP AND PH                                   UFP07270
C                                                                       UFP07280
      DO 11 J=1,3                                                       UFP07290
      IF(K.EQ.1) A(1,J)=A(1,J)*BNFB                                     UFP07300
      IF(K.EQ.NHMAX) GO TO 12                                           UFP07310
      IF(K.GT.1) A(K,J)=A(K,J)*ANFB                                     UFP07320
      IF(K.GT.1) B(K,J)=B(K,J)*ANFB                                     UFP07330
      GO TO 13                                                          UFP07340
 12   A(K,J)=A(K,J)*BNFB                                                UFP07350
      B(K,J)=0.0                                                        UFP07360
 13   AMP(K,J)=SQRT(A(K,J)**2+B(K,J)**2)                                UFP07370
      PH(K,J)=ATAN2(A(K,J),B(K,J))*57.296                               UFP07380
      IF(PH(K,J).LT.0.0) PH(K,J)=PH(K,J)+360.0                          UFP07390
      PH(K,J)=PH(K,J)/NBLADE                                            UFP07400
      PH(1,J)=0.0                                                       UFP07410
      AMP(1,J)=A(1,J)                                                   UFP07420
 11   CONTINUE                                                          UFP07430
      RETURN                                                            UFP07440
      END                                                               UFP07450
```

```
      SUBRDUTINE PIEDAT(ISTOP)
C     **********************************************************
C     *                                                        *
C     *   SUBROUTINE PIEOAT GENERATES OATA FROM INPUT TO       *
C     *   AXIAL VELOCITY DUTPUT TO BE USEO AS INPUT FOR        *
C     *   UNSTEAOY PIEWAKE.  IT USES BLADE GRID GEOMETRY       *
C     *   TD CALCULATE NSR FIELD POINT AXIAL TIME-AVERAGED     *
C     *   VELDCITIES AT FIVE RADII ALDNG THE LEADING EDGE.     *
C     *   IT THEN CALCULATES NSR FIELD POINT AXIAL TIME-       *
C     *   AVERAGED VELOCITIES AT 1.1 TIMES PROPELLER RAOIUS*
C     *                                                        *
C     *   N          => CHORDWISE GRID CDORDINATE              *
C     *   M          => SPANWISE GRID CDORDINATE               *
C     *   K,KK       => ANGLE INCREMENTS + 1                   *
C     *   ISTDP=1    => STDP FPUV UPDN RETURN TD MAIN PROG     *
C     *   NSR        => NUMBER TIME STEPS IN REVDLUTIDN        *
C     *   NR         => RADIUS NUMBER                          *
C     *   XFP,YBAR,ZBAR,RFP,TZFP                               *
C     *              => X,Y,Z,RAOIAL,THETA FIELO PT COORD      *
C     *   THETA(1-5)=> ANGLES 5 TIME STEPS BEFDRE ZERO         *
C     *                 BLAOE ANGLE                            *
C     *   THETA(6 - NSR+5) => ANGLES IN ONE BLAOE REV.         *
C     *   THETA(NSR+6 - NSR+20) => ANGLES 15 TIME STEPS        *
C     *                 AFTER ZERD BLAO ANGLE.                 *
C     *   VX(K)      => AXIAL VELDCITY AT ANGLE THETA(K)       *
C     *   UA(KK,M)   => SMOOTH AXIAL VELOCITY AT RADIUS M,     *
C     *                 ANGLE KK                               *
C     *                                                        *
C     **********************************************************
C
C
C
C *** CDMMON TD CONTAIN GEOMETRIC PARAMETERS
C
      COMMON / GEDCDM / MM,MN,NN,NW,NTORL,X(12,10),Y(12,10),Z(12,10),
     &   XTIP(10,11),YTIP(10,11),ZTIP(10,11),XH(101,7),YH(101,7),
     &   ZH(101,7),XW(21,10),YW(21,10),ZW(21,10),RH
C
C *** COMMON TD CONTAIN SINGULARITY STRENGTHS
C
      CDMMDN / SINCDM / GT(10,10,60),G8(10,9,60),GTW(10),SB(10,9),GTV
     &  ,GAM(9,60),GMEAN(9),DKFW(20),HBWX,HBWY,HBWZ
C
C *** COMMDN TD CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, ANO VSR
C
      COMMON / FPCDM / NBLAOE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM
C
C *** COMMON TD CONTAIN VELOCITY OATA
C
      COMMON / VELCOM / VIX(60),VIT(60),VIR(60),U(60,3),NSAMP
C
C *** COMMON TO CDNTAIN HARMDNICS COEFFICIENTS
```

```
C                                                                       UFP07990
      COMMON / HARCOM / A(15,3),B(15,3),AMP(15,3),PH(15,3),NH           UFP08000
C                                                                       UFP08010
C                                                                       UFP08020
      DIMENSION THETA(80),VX(80),UA(60,9),RWK(9)                        UFP08030
      N=1                                                               UFP08040
      NR=1                                                              UFP08050
C                                                                       UFP08060
C *** CALCULATE VELOCITIES AT FIVE BLADE GRID RADII                     UFP08070
C                                                                       UFP08080
 102  DO 300 M=3,9,2                                                    UFP08090
      NR=NR+1                                                           UFP08100
C                                                                       UFP08110
C *** CALCULATE VELOCITIES AT NSR ANGLES FOR EACH RADIUS                UFP08120
C                                                                       UFP08130
      DO 200 KK=1,NSR                                                   UFP08140
C                                                                       UFP08150
C *** FIELD POINT CENTERED ON BLADE GRID                                UFP08160
C                                                                       UFP08170
      CALL FPGRID(X,Y,Z,N,M,KK,12)                                      UFP08180
      IF (M.EQ.3) X1=XFP                                                UFP08190
      KK15=KK+15                                                        UFP08200
      THETA(KK15)=TZFP                                                  UFP08210
      RWK(NR)=RFP                                                       UFP08220
      DTFP=360./FLOAT(NSR)                                              UFP08230
C                                                                       UFP08240
C *** CALCULATE AXIAL VELOCITY FOR FIELD POINT AT RADIUS RWK(NR) AND    UFP08250
C *** ANGLE THETA(KK15).  STORE THIS VELOCITY IN VX.                    UFP08260
C                                                                       UFP08270
 200  CALL AMPCAL(VX(16),KK)                                            UFP08280
      NSR15=NSR+15                                                      UFP08290
C                                                                       UFP08300
C *** STORE SMOOTHED VELOCITIES IN NR-TH COLUMN OF UA                   UFP08310
C                                                                       UFP08320
 300  CALL SMOOTH(NSR,NR,VX,THETA,UA)                                   UFP08330
C                                                                       UFP08340
C *** CALCULATE NSR VELOCITIES AT PROPELLER HUB                         UFP08350
C                                                                       UFP08360
      NRHUB=1                                                           UFP08370
      XFP=1.5*X1                                                        UFP08380
      RWK(NRHUB)=RH                                                     UFP08390
C                                                                       UFP08400
C *** STEP THROUGH NSR ANGLES AT HUB                                    UFP08410
C                                                                       UFP08420
      DO 100 KK=1,NSR                                                   UFP08430
      KK15=KK+15                                                        UFP08440
      THETA(KK15)=360.*FLOAT(KK-1)/FLOAT(NSR)                           UFP08450
      TZFP=THETA(KK15)                                                  UFP08460
 100  CALL AMPCAL(VX(16),KK)                                            UFP08470
      NSR15=NSR+15                                                      UFP08480
C                                                                       UFP08490
C *** NOW STORE SMOOTHED HUB VELOCITIES IN FIRST COLUMN OF UA           UFP08500
C                                                                       UFP08510
```

```
      CALL SMOOTH(NSR,NRHUB,VX,THETA,UA)                                UFP08520
C                                                                       UFP08530
C *** NOW CALCULATE VELOCITIES AT A NEXT RADII OUTSIDE THE PROPELLER    UFP08540
C *** DISK.  USE XFP OF PROPELLER GRID CASE OF N=1, M=9.                UFP08550
C                                                                       UFP08560
C &&& DIAGNOSTICS &&&&&&&&&&&&&&&&                                      UFP08570
C                                                                       UFP08580
C     GOTO 101                                                          UFP08590
C                                                                       UFP08600
C *** CALCULATE NSR VELOCITIES AT 1.1,1.4, AND 1.7 PROP RADII           UFP08610
C                                                                       UFP08620
      DO 500 KKK=1,7,3                                                  UFP08630
      RFP=1.0+FLOAT(KKK)/10.                                            UFP08640
      NR=NR+1                                                           UFP08650
      RWK(NR)=RFP                                                       UFP08660
C                                                                       UFP08680
C *** CALCULATE NSR VELOCITIES AT THESE RADII                           UFP08690
C                                                                       UFP08700
      DO 400 KK=1,NSR                                                   UFP08710
      KK15=KK+15                                                        UFP08720
      THETA(KK15)=360.*FLOAT(KK-1)/FLOAT(NSR)                           UFP08730
      TZFP=THETA(KK15)                                                  UFP08740
  400 CALL AMPCAL(VX(16),KK)                                            UFP08750
C                                                                       UFP08760
C *** NOW STORE SMOOTHED VELOCITIES IN NR-TH COLUMN OF UA               UFP08770
C                                                                       UFP08780
  500 CALL SMOOTH(NSR,NR,VX,THETA,UA)                                   UFP08790
C                                                                       UFP08800
C *** WRITE(OUTPUT IN FORMAT FOR UNSTEADY PIEWAKE                       UFP08810
C                                                                       UFP08820
  901 FORMAT(8F10.5)                                                    UFP08830
  902 FORMAT(16I5)                                                      UFP08840
  101 WRITE(8,902) NR                                                   UFP08850
      WRITE(8,901) (RWK(M),M=1,NR)                                      UFP08860
      WRITE(8,901) ((UA(KK,M),KK=1,NSR),M=1,NR)                         UFP08870
C                                                                       UFP08880
C *** STOP FPUV UPON RETURN TO MAIN PROGRAM                             UFP08890
C                                                                       UFP08900
      ISTOP=1                                                           UFP08910
      RETURN                                                            UFP08920
      END                                                               UFP08930
```

```
      SUBROUTINE AMPCAL(VX,KK)                                          UFP08940
C     *****************************************************             UFP08950
C     *                                                   *             UFP08960
C     *   SUBROUTINE AMPCAL WILL TAKE FIELD POINT GEOMETRY *            UFP08970
C     *   CALCULATED IN PIEDAT AND CALCULATE THE INDUCED   *            UFP08980
C     *   TIME-AVERAGED AXIAL VELOCITY.  IT WILL STORE THAT*            UFP08990
C     *   VELOCITY IN VX(KK).                              *            UFP09000
C     *                                                   *             UFP09010
C     *   NSR        => NUMBER OF ANGLES PER REVOLUTION    *            UFP09020
C     *   K          => ANGLE INCREMENTS + 1               *            UFP09030
C     *   VX(KK)     => WHERE IME-AVERAGED AXIAL VEL STORED *           UFP09040
C     *   AMP(1,1)   => ZEROETH-HARMONIC AXIOAL VELOCITY   *            UFP09050
C     *                 COMPUTED BY FOUCAL                 *            UFP09060
C     *                                                   *             UFP09070
C     *****************************************************             UFP09080
C                                                                       UFP09090
C                                                                       UFP09100
C                                                                       UFP09110
C *** COMMON TO CONTAIN GEOMETRIC PARAMETERS                            UFP09120
C                                                                       UFP09130
      COMMON / GEOCOM / MM,MN,NN,NW,NTORL,X(12,10),Y(12,10),Z(12,10),   UFP09140
     &   XTIP(10,11),YTIP(10,11),ZTIP(10,11),XH(101,7),YH(101,7),       UFP09150
     &   ZH(101,7),XW(21,10),YW(21,10),ZW(21,10),RH                     UFP09160
C                                                                       UFP09170
C *** COMMON TO CONTAIN SINGULARITY STRENGTHS                           UFP09180
C                                                                       UFP09190
      COMMON / SINCOM / GT(10,10,60),GB(10,9,60),GTW(10),SB(10,9),GTV   UFP09200
     &  ,GAM(9,60),GMEAN(9),DKFW(20),HBWX,HBWY,HBWZ                     UFP09210
C                                                                       UFP09220
C *** COMMON TO CONTAIN FIELD POINT DATA, NBLADE, IDENT, NSR, AND VSR   UFP09230
C                                                                       UFP09240
      COMMON / FPCOM / NBLADE,NSR,XFP,RFP,TZFP,YBAR,ZBAR,               UFP09250
     &   DBLADE,DTFP,IDENT(18),TFP(60),VSR,IPLOT,DIAM                   UFP09260
C                                                                       UFP09270
C *** COMMON TO CONTAIN VELOCITY DATA                                   UFP09280
C                                                                       UFP09290
      COMMON / VELCOM / VIX(60),VIT(60),VIR(60),U(60,3),NSAMP           UFP09300
C                                                                       UFP09310
C *** COMMON TO CONTAIN HARMONICS COEFFICIENTS                          UFP09320
C                                                                       UFP09330
      COMMON / HARCOM / A(15,3),B(15,3),AMP(15,3),PH(15,3),NH           UFP09340
C                                                                       UFP09350
      DIMENSION VX(1)                                                   UFP09360
C                                                                       UFP09370
C *** RUN THROUGH AND FPSTEP FOR EACH TIME STEP                         UFP09380
C                                                                       UFP09390
      DO 100 K=1,NSR                                                    UFP09400
C                                                                       UFP09410
C *** FPSTEP CALCULATES VELOCITY INDUCED AT A FIELD POINT BY ONE        UFP09420
C *** BLADE AT TIME STEP K                                              UFP09430
C                                                                       UFP09440
  100 CALL FPSTEP(K)                                                    UFP09450
C                                                                       UFP09460
```

```
C *** SUMVEL SUMS FIELD POINT VELOCITIES INDUCED BY EACH BLADE          UFP09470
C                                                                       UFP09480
      CALL SUMVEL                                                       UFP09490
C                                                                       UFP09500
C *** FOUCAL GENERATES HARMONIC COEFFICIENTS FOR INDUCED FIELD POINT    UFP09510
C *** VELOCITY AS PROPELLER ROTATES.                                    UFP09520
C                                                                       UFP09530
      CALL FOUCAL                                                       UFP09540
C                                                                       UFP09550
C *** STORE TIME-AVERAGED (ZERO-HARMONIC) VELOCITY IN VX                UFP09560
C                                                                       UFP09570
      VX(KK)=AMP(1,1)                                                   UFP09580
      RETURN                                                            UFP09590
      END                                                               UFP09600
```

```
      SUBROUTINE SMOOTH(NSR,NR,VX,THETA,UA)
C     *****************************************************
C     *                                                   *
C     *   SUBROUTINE SMOOTH READS INDUCED VELOCITIES AT ONE*
C     *   RADIUS FROM PIEDAT.  IT CALCULATES BY           *
C     *   INTERPOLATION A SERIES OF NSR EVENLY SPACED     *
C     *   AXIAL INDUCED VELOCITIES SUITABLE FOR INPUT INTO *
C     *   UNSTEADY PIEWAKE.                               *
C     *                                                   *
C     *   NSR        => NUMBER OF ANGLES PER REVOLUTION   *
C     *   NR         => RADIUS NUMBER 1=HUB, ETC.         *
C     *   VX(1 - 5) => RAW VELOCITIES 5 STEPS BEFORE ZERO  *
C     *                BLADE ANGLE.                       *
C     *   VX(6 - NSR+5) => RAW VELOCITIES IN ONE REVOLUTION*
C     *   VX(NSR+6 - NSR+20) => RAW VELOCITIES 15 TIME    *
C     *                TIME STEPS AFTER ONE REVOLUTION    *
C     *   THETA(K)   => ANGLE OF VELOCITY VX(K)           *
C     *   UA(K,M)    => SMOOTH VEL AT RADIUS M, ANGLE K   *
C     *   TH(K)      => EVENLY SPACED ANGLES OF UA(K,M)   *
C     *   NIN        => NUMBER OF ANGLES FED TO UGLYOK    *
C     *   KPOS       => K OF FIRST POS. ANGLE IN THETA(K) *
C     *   KSTART,KSTOP => FIRST AND LAST INDEX OF THETA(K) *
C     *                AND VX(K) SENT TO UGLYOK           *
C     *   AVX        => SPLINE CURVE COEFFICIENTS OF VX   *
C     *                                                   *
C     *****************************************************
C
C
      DIMENSION VX(80),UA(60,9),THETA(80),TH(60),AVX(250)
C
C *** COPY LAST 15 STEPS IN REV INTO FIRST 15 ENTRIES IN VX AND THETA
C
      DO 100 K=1,15
      KNSR=K+NSR
      THETA(K)=THETA(KNSR)-360.
 100  VX(K)=VX(KNSR)
C
C *** COPY FIRST 5 TIME STEPS IN REV INTO 5 ENTRIES AFTER REVOLUTION
C
      DO 200 K=16,20
      KNSR=K+NSR
      THETA(KNSR)=THETA(K) + 360.
 200  VX(KNSR)=VX(K)
C
C *** SET UP EVENLY SPACED ANGLES
C
      DO 300 K=1,NSR
 300  TH(K)=360.*FLOAT(K-1)/FLOAT(NSR)
 901  FORMAT(8F10.5)
 902  FORMAT(16I5)
C
C *** FIND FIRST POSITIVE ANGLE, SAVE INDEX
C
```

```
      DO 400 K=1,NSR                                                    UFP10140
      KPOS=K                                                            UFP10150
      IF (THETA(K).GT.O.) GOTO 500                                      UFP10160
  400 CONTINUE                                                          UFP10170
  500 KSTART=KPOS-1                                                     UFP10180
      NSR2=NSR/2                                                        UFP10190
      KSTOP=KPOS+NSR2+1                                                 UFP10200
      NIN=KSTOP-KSTART+1                                                UFP10210
C                                                                       UFP10220
C *** FINE SPLINE COEFFICIENTS AND SMOOTH VEL FOR FIRST 180 DEGREES     UFP10230
C                                                                       UFP10240
      CALL UGLYDK(NIN,1,1,THETA(KSTART),VX(KSTART),O.,O.,AVX)           UFP10250
      CALL EVALDK(NIN,NSR2,THETA(KSTART),TH(1),UA(1,NR),AVX)            UFP10260
C                                                                       UFP10270
C *** FIND SPLINE COEFFICIENTS AND SMOOTH VEL FOR LAST 180 DEGREES      UFP10280
C                                                                       UFP10290
      KSTART=KSTOP-2                                                    UFP10300
      KSTOP=KSTART+NIN-1                                                UFP10310
      NSR2P=NSR2+1                                                      UFP10320
      CALL UGLYDK(NIN,1,1,THETA(KSTART),VX(KSTART),O.,O.,AVX)           UFP10330
      CALL EVALDK(NIN,NSR2,THETA(KSTART),TH(NSR2P),UA(NSR2P,NR),AVX)    UFP10340
      RETURN                                                            UFP10350
      END                                                               UFP10360
```

** APPENDIX XI **

PIEWAKE Program Listing

```
C     THIS PROGRAM COMPUTES EFFECTIVE WAKES BY DNV METHOD
C
C     ******************************************************
C     *                                                    *
C     *   THE ORIGINAL PIEWAKE WAS WRITTEN BY R. VAN HOUTEN*
C     *   IN ABOUT 1981.  IT IS BASED ON THE DET NORSKE    *
C     *   VERITAS METHOD OF COMPUTING EFFECTIVE WAKE.  THIS*
C     *   PROGRAM HAS BEEN MODIFIED BY R. JAMISON IN APRIL.*
C     *   1982, TO ACCOMODATE UNSTEADY PROPELLER-INDUCED   *
C     *   AXIAL VELOCITIES FROM UFPV.  BASICALLY, BOTH     *
C     *   PROGRAMS DIVIDE THE WAKE INTO A NUMBER OF PIE-   *
C     *   SHAPED SEGMENTS.  EACH SEGMENT IS ASSUMED TO     *
C     *   CONTRACT IN AN AXISYMMETRIC MANNER AS GIVEN BY   *
C     *   T. HUANG.                                        *
C     *                                                    *
C     *   NP          => NUMBER OF PIE-SHAPED SEGMENTS     *
C     *   NX          => NUMBER OF NOMINAL WAKE RADII      *
C     *   RX(M)       => NOMINAL WAKE RADIUS M (UP TO 11)  *
C     *   VO(K,M)     => NOMINAL AXIAL VELOCITY, SEG K, RAD M*
C     *   VT(K,M)     => NOM TANG VEL, SEG K, RAD M        *
C     *   VR(K,M)     => NOM RAD VEL, SEG K, RAD M         *
C     *   VTE,VRE     => VT & VR INTERPOLATED AT EFF WAKE RAD*
C     *   UASEG(M)    => UA IN A PIE-SEGMENT.  ALSO USED TO *
C     *                  STORE A PIE-SEGMENT OF VT         *
C     *   UX(M)       => VO IN A PIE-SEGMENT.  ALSO USED TO *
C     *                  STORE A PIE-SEGMENT OF VR         *
C     *   AUX,AUASEG=> SPLINE COEFF. FOR VR,VT, RESP.      *
C     *   RE(M)       => RADII OF EFFECTIVE WAKE VELOCITIES *
C     *   THETA(K)    => ANGLE IN DEGREES OF PIE SEG K     *
C     *   NA          => NUMBER OF PROP-IND. AXIAL VEL RADII *
C     *   RA(M)       => PROP-IND AXIAL VEL RADIUS M       *
C     *   UA(K,M)     => PROP IND AXIAL VEL, SEG K, RAD M  *
C     *   VE(K,M)     => EFFECTIVE WAKE VEL, SEG K, RAD M  *
C     *   VL          => AVERAGE VELOCITY                  *
C     *   VLAVN       => VOLUMETRIC AVERAGE NOMINAL VELOCITY *
C     *   VLAVE       => VOLUMETRIC AVERAGE EFFECTIVE VELOCITY
C     *   BLOCK       => EFFECTIVE BLOCKAGE                *
C     *                                                    *
C     ******************************************************
      DIMENSION VO(60,11),RX(11),UX(60),RA(11),UA(60,11),UASEG(60)
      DIMENSION RE(11), VR(60,11),VT(60,11),AUASEG(44),AUX(44),VTE(11)
      DIMENSION THETA(60),VE(60,11),IDENT(18),UERX(60),VRE(11)
      COMMON RX,UX,RA,UASEG,UERX,RE
  110 FORMAT(8F10.5)
  103 FORMAT(/' AVERAGE VELOCITY =',F5.3)
  113 FORMAT(/' VOLUMETRIC AVERAGE NOMINAL VELOCITY =',F5.3)
  114 FORMAT(/' VOLUMETRIC AVERAGE EFFECTIVE VELOCITY =',F5.3)
  115 FORMAT(/' EFFECTIVE BLOCKAGE =',F7.3,' PERCENT')
  111 FORMAT(18A4)
  210 FORMAT(16I5)
  200 FORMAT(8F10.5)
C
C *** READ IN NOMINAL WAKE DATA.  SAVE RADIAL AND TANGENTIAL
```

```
C     VELOCITIES.  THEY ARE NOT NEEDED FOR EFFECTIVE WAKE
C     CALCULATIONS BUT WILL BE ECHOED FOR WKPRDC.
C
      READ(10,111) (IDENT(I),I=1,18)
      READ(10,210) NX
      READ(10,210) NP
      READ(10,200)(RX(M),M=1,NX)
      DO 20 M=1,NX
      READ(10,200)((THETA(K),VD(K,M),VT(K,M),VR(K,M)),K=1,NP)
   20 CONTINUE
C
C *** READ IN PROPELLER-INDUCED VELOCITIES FROM EITHER FPUV
C     OR A SEPARATE DATA FILE
C
 900  FORMAT(/' DO YOU WISH TO USE STEADY OR UNSTEADY VELOCITY PROFILE?'
     &  /' ENTER 1 FOR STEADY, 2 FOR UNSTEADY')
   8  WRITE(6,900)
      READ(5,*) ISTED
      IF (ISTED.GT.2.OR.ISTED.LT.1) GOTO 8
      READ(8,210) NA
      READ(8,200) (RA(I),I=1,NA)
      RH=RA(1)
      RPROP=1.
      IF(ISTED.EQ.1) GOTO 22
      READ(8,200) ((UA(KK,M),KK=1,NP),M=1,NA)
      GOTO 92
  22  READ(8,200) (UA(1,M),M=1,NA)
      DO 21 KK=2,NP
      DO 21 M=1,NA
  21  UA(KK,M)=UA(1,M)
C
C *** EXTRAPOLATE NOMINAL WAKE DATA TO HUB
C
  92  WRITE(6,100)
  100 FORMAT(/' DO YOU WISH TO EXTRAPOLATE VELOCITY DATA TO THE HUB?'/
     1  ' ENTER 2 FOR NO, 1 FOR LINEAR, 0 FOR CONSTANT')
      READ(5,*)IXTRAP
      NX1=NX
      IF(IXTRAP.GT.1)GO TO 28
C
C *** LINEAR EXTRAPOLATION OF NOMINAL WAKE DATA TO HUB.
C
      NX=NX+1
      DO 25 K=1,NP
      DO 23 M=1,NX1
      MR=NX-M
      MRP1=MR+1
      VT(K,MRP1)=VT(K,MR)
      VR(K,MRP1)=VR(K,MR)
   23 VD(K,MRP1)=VD(K,MR)
   25 VD(K,1)=VD(K,2)+IXTRAP*(VD(K,2)-VD(K,3))/(RX(1)-RX(2))*(RH-RX(1))
      DO 26 M=1,NX1
      MR=NX-M
```

```
      MRP1=MR+1                                                         PIE01070
   26 RX(MRP1)=RX(MR)                                                   PIE01080
      RX(1)=RH                                                          PIE01090
   28 VL=0.                                                             PIE01100
      VLAVN=0.                                                          PIE01110
      VLAVE=0.                                                          PIE01120
C                                                                       PIE01130
C *** CYCLE THROUGH EACH PIE SEGMENT                                    PIE01140
C                                                                       PIE01150
      DO 50 K=1,NP                                                      PIE01160
      DO 30 M=1,NX                                                      PIE01170
      UASEG(M)=UA(K,M)                                                  PIE01180
   30 UX(M)=VO(K,M)                                                     PIE01190
C                                                                       PIE01200
C *** WKMOD PERFORMS THE HUANG MODIFICATION ON ONE PIE-SEGMENT          PIE01210
C                                                                       PIE01220
      CALL WKMOD(NX,NX,NA,1,RPROP,VOL,VOLAVN,VOLAVE)                    PIE01230
C                                                                       PIE01240
C *** INTERPOLATE VT AND VR AT EFFECTIVE WAKE RADII                     PIE01250
C                                                                       PIE01260
      DO 60 M=1,NX                                                      PIE01270
      UASEG(M)=VT(K,M)                                                  PIE01280
  60  UX(M)=VR(K,M)                                                     PIE01290
      CALL UGLYDK(NX,1,1,RX,UASEG,0.,0.,AUASEG)                         PIE01300
      CALL UGLYDK(NX,1,1,RX,UX,0.,0.,AUX)                               PIE01310
      CALL EVALDK(NX,NX,RX,RE,VTE,AUASEG)                               PIE01320
      CALL EVALDK(NX,NX,RX,RE,VRE,AUX)                                  PIE01330
      DO 70 M=1,NX                                                      PIE01340
      VT(K,M)=VTE(M)                                                    PIE01350
  70  VR(K,M)=VRE(M)                                                    PIE01360
C                                                                       PIE01370
C *** ACCUMULATE VOLUMETRIC AVERAGE DATA                                PIE01380
C                                                                       PIE01390
      VL=VL+VOL                                                         PIE01400
      VLAVN=VLAVN+VOLAVN                                                PIE01410
      VLAVE=VLAVE+VOLAVE                                                PIE01420
      DO 40 M=1,NX                                                      PIE01430
   40 VE(K,M)=UERX(M)                                                   PIE01440
   50 CONTINUE                                                          PIE01450
C                                                                       PIE01460
C *** COMPUTE VOLUMETRIC AVERAGES                                       PIE01470
C                                                                       PIE01480
      VL=VL/NP                                                          PIE01490
      VLAVN=VLAVN/NP                                                    PIE01500
      VLAVE=VLAVE/NP                                                    PIE01510
      BLOCK=(1.-VLAVE/VLAVN)*100.                                       PIE01520
C                                                                       PIE01530
C *** WRITE EFFECTIVE WAKE DATA IN FILE 11 TO BE PROCESSED BY WKPROC    PIE01540
C                                                                       PIE01550
      WRITE(11,111) (IDENT(I),I=1,18)                                   PIE01560
      WRITE(11,210) NX                                                  PIE01570
      WRITE(11,210) (NP,M=1,NX)                                         PIE01580
      WRITE(11,200)(RE(M),M=1,NX)                                       PIE01590
```

```
      DO 120 M=1,NX                                                     PIE01600
      WRITE(11,200)((THETA(K),VE(K,M),VT(K,M),VR(K,M)),K=1,NP)          PIE01610
  120 CONTINUE                                                          PIE01620
C                                                                       PIE01630
C *** WRITE AVERAGES TO TERMINAL                                        PIE01640
C                                                                       PIE01650
      WRITE(6,103)VL                                                    PIE01660
      WRITE(6,113)VLAVN                                                 PIE01670
      WRITE(6,114)VLAVE                                                 PIE01680
      WRITE(6,115)BLOCK                                                 PIE01690
      STOP                                                              PIE01700
      END                                                               PIE01710
```

```
      SUBROUTINE WKMOD(N,NX,NA,IOW,RPROP,VOL,VOLAVN,VOLAVE)             PIE01720
C                                                                       PIE01730
C     ******************************************************            PIE01740
C     *                                                    *            PIE01750
C     *    SUBROUTINE WKMOD CALCULATES THE THOMAS HUANG    *            PIE01760
C     *    CONTRACTION OF THE NOMINAL AND INDUCED WAKES    *            PIE01770
C     *    PASSED TO IT THROUGH COMMON.  IT ALSO COMPUTES  *            PIE01780
C     *    VOLUMETRIC AVERAGES.                            *            PIE01790
C     *                                                    *            PIE01800
C     *    NX          => NUMBER OF NOMINAL RADII          *            PIE01810
C     *    RH          => RADIUS OF HUB                    *            PIE01820
C     *    RT          => RADIUS OF OUTERMOST NOMINAL VELOCITY*          PIE01830
C     *    UX,UXR      => NOMINAL WAKE VELOCITIES          *            PIE01840
C     *    UE,UEXR     => EFFECTIVE WAKE VELOCITIES        *            PIE01850
C     *    UP          => APPARENT WAKE = EFF. WAKE + IND. WAKE         PIE01860
C     *    UA,UAR      => INDUCED VELOCITIES               *            PIE01870
C     *    IOW = 1     => NO TUNNEL CORRECTIONS            *            PIE01880
C     *    ITER        => OPEN WATER ITERATION INDEX       *            PIE01890
C     *    ITERAT      => HUANG CORRECTION FACTOR INDEX    *            PIE01900
C     *    B,C,D,F     => HUANG CORRECTION FACTORS. SEE PAPER *         PIE01910
C     *    AUA,AUX,AUE=> SPLINE COEFF. FOR INDUCED, NOMINAL,*           PIE01920
C     *                  AND EFFECTIVE VELOCITY FIELDS, RESP *          PIE01930
C     *    RPROP       => RADIUS OF PROPELLER              *            PIE01940
C     *    RE(M)       => SPECIFIED EFFECTIVE WAKE RADII   *            PIE01950
C     *    VOL         => AVERAGE VELOCITY                 *            PIE01960
C     *    VLAVN       => VOLUMETRIC AVERAGE NOMINAL VELOCITY *         PIE01970
C     *    VLAVE       => VOLUMETRIC AVERAGE EFFECTIVE VELOCITY         PIE01980
C     *                                                    *            PIE01990
C     ******************************************************            PIE02000
      DIMENSION RX(11),R(11),RA(11),UX(60),UXR(60),UA(60),UP(60),UE(60) PIE02010
      DIMENSION RE(11),UAR(60),RP(11)                                   PIE02020
      DIMENSION AUX(240),AUA(240)                                       PIE02030
      DIMENSION AUE(240),UERX(60)                                       PIE02040
      COMMON RX,UX,RA,UA,UERX,RE                                        PIE02050
C                                                                       PIE02060
C *** USE R(I) FOR NOMINAL RADII                                        PIE02070
C                                                                       PIE02080
      N=NX                                                              PIE02090
      N1=N-1                                                            PIE02100
      RH=RX(1)                                                          PIE02110
      RT=RX(NX)                                                         PIE02120
C                                                                       PIE02130
C *** INTERPOLATE TO FIND NOMINAL VELOCITIES AT NOMINAL RADII           PIE02140
C                                                                       PIE02150
      CALL UGLYDK(NX,1,1,RX,UX,0.,0.,AUX)                               PIE02160
      DO 10 I=1,N                                                       PIE02170
C  10 R(I)=(RT-RH)/(N-1)*(I-1)+RH                                       PIE02180
   10 R(I)=RX(I)                                                        PIE02190
      CALL EVALDK(NX,N,RX,R,UXR,AUX)                                    PIE02200
C                                                                       PIE02210
C *** INTERPOLATE TO FIND INDUCED VELOCITIES AT NOMINAL RADII           PIE02220
C                                                                       PIE02230
      CALL UGLYDK(NA,1,1,RA,UA,0.,0.,AUA)                               PIE02240
```

```
      CALL EVALOK(NA,N,RA,R,UAR,AUA)
  901 FORMAT(8F10.5)
C
C *** OUTSIDE PROPELLER DISK, NOMINAL WAKE = EFFECTIVE WAKE
C
      IUPDAT=0
      ITER=1
      ITERAT=1
      UE(N)=UXR(N)
   20 UP(N)=UE(N)+UAR(N)
C
C *** ESTIMATE EFF. WAKE MARCHING FROM OUTSIDE TO HUB
C
      DO 30 I=1,N1
      K=N-I
      UE(K)=SQRT((UE(K+1)+(UAR(K+1)+UAR(K))/2.)**2+UXR(K)**2-UXR(K+1)**2
     1)-(UAR(K+1)+UAR(K))/2.
      UP(K)=UE(K)+UAR(K)
   30 CONTINUE
C
C *** COMPUTE HUANG'S CORRECTION FACTORS
C
      RP(1)=RH
      VOL=0.
      DO 40 K=1,N1
      B=2*UP(K+1)+UP(K)
      C=-RP(K)*(UP(K+1)-UP(K))
      F=(R(K+1)**2-R(K)**2)*(2*UXR(K+1)+UXR(K))-R(K)*(UXR(K+1)-UXR(K))*(
     1R(K+1)-R(K))
      VOL=VOL+F
      D=-RP(K)**2*(UP(K+1)+2*UP(K))-F
C
C *** FIND RADIUS FOR (K+1)ST APPARENT WAKE
C
      IF ((C*C-4*B*D).LT.0.) GOTO 80
   40 RP(K+1)=(-C+SQRT(C**2-4*B*D))/2./B
      IF(ABS(RP(N)-RT).LT.(.0001*RT))GO TO 60
      IUPDAT=0
C
C *** IOW=0 => TUNNEL CORRECTIONS
C        =1 => OPEN WATER (I.E., INFINITE FLUID)
C
      IF(IOW.EQ.1) GO TO 60
      IF (ITER.NE.1)GO TO 50
      ITER=2
      UENOLD=UE(N)
      RPNOLD=RP(N)
      UE(N)=UE(N)*.98
      GO TO 20
   50 CONTINUE
      ITER=ITER+1
      IF(ITER.GT.10)GO TO 80
      UEN=UE(N)
```

```
      UE(N)=UE(N)+(RT-RP(N))/(RP(N)-RPNOLD)*(UE(N)-UENOLD)
      RPNOLD=RP(N)
      UENOLD=UEN
      GO TO 20
C
C *** ITERATE ON HUANG CORRECTION FACTORS, 3 TIMES FOR OPEN WATER
C
   60 IF(IUPDAT.EQ.1)GO TO 70
      IF(IDW.EQ.1.AND.ITERAT.EQ.5)GO TO 70
      ITER=1
      ITERAT=ITERAT+1
      IF(ITERAT.GT.10)GO TO 80
      IUPDAT=1
C
C *** FIND INDUCED VELOCITIES AT APPARENT WAKE RADII
C
      CALL EVALDK(NA,N,RA,RP,UAR,AUA)
      GO TO 20
   70 CONTINUE
C
C *** CALCULATE EFFECTIVE WAKE RADII
C
      NXM1=NX-1
      DO 100 M=1,NX
  100 RE(M)=RH+(RPROP-RH)*FLOAT(M-1)/FLOAT(NXM1)
C
C *** IF OUTERMOST APPARENT RADIUS IS INSIDE PROPELLER DISK, THEN
C     LINEARLY EXTRAPOLATE TO PROPELLER RADIUS
C
      IF (RP(N).GE.RPROP) GOTO 110
      UE(N)=UE(N1) + (UE(N)-UE(N1))*(RPROP-R(N1))/(RP(N)-RP(N1))
      RP(N)=RPROP
C
C *** INTERPOLATE EFFECTIVE WAKE VELOCITIES AT SPECIFIED RADII
C
  110 CALL UGLYDK(N,1,1,RP,UE,0.,0.,AUE)
      CALL EVALDK(N,NX,RP,RE,UERX,AUE)
C
C *** COMPUTE VOL = AVERAGE VELOCITY
C
      VOL=VOL/3./(RT**2-RH**2)
C
C *** COMPUTE VOLUMETRIC AVERAGE NOMINAL VELOCITY
C
      CALL INTEDK(NX,RX,RH,RPROP,YDX,XYDX,XXYDX,AUX)
      VOLAVN=XYDX*2/(RPROP**2-RH**2)
C
C *** COMPUTE VOLUMETRIC AVERAGE EFFECTIVE VELOCITY
C
      CALL INTEDK(N,RP,RH,RPROP,YDX,XYDX,XXYDX,AUE)
      VOLAVE=XYDX*2./(RPROP**2-RH**2)
      RETURN
C
```

```
C *** TUNNEL CORRECTION OR HUANG CORRECTION ITERATIONS FAIL TO CONVERGE PIE03310
C                                                                       PIE03320
   80  STOP                                                             PIE03330
       END                                                              PIE03340
```

REFERENCES

(1) Boswell, Robert J., "Design, Cavitation Performance, and Open-Water Performance of a Series of Research Skewed Propellers", Naval Research and Development Center Report No. 3339, 1971

(2) Valentine, D.T., and Chase, A., "Highly Skewed Propeller Design for a Naval Auxiliary Oiler (AO-177)", Naval Research and Development Center Report No. SPD-544-12, 1976

(3) Hendrican, A., and Remmers, K., "Powering and Cavitation Performance for a Naval Fleet Oiler, AO-177 Class (Model 5326 with Propeller 4677)", Naval Research and Development Center Report No. SPD-544-14, 1976

(4) Burrill, L.C. and Emerson, A., "Propeller Cavitation: Further Tests on 16-inch Propellers in the Kings College Cavitation Tunnel", Northeast Coast Institute of Engineers and Shipbuilders, Vol. 79, pp. 295-320, 1962-63

(5) Kerwin, J.E., and Lee, C.-S., "Prediction of Stedy and Unsteady Marine Propeller Performance by Numerical Lifting-Surface Theory", SNAME Transactions, Vol. 86, pp. 218-253, 1978

(6) Min, K.-S., "Numerical and Experimental Methods for

the Prediction of Field Point Velocities Around Propeller Blades", Department of Ocean Engineering of Massachusetts Institute of Technology Report No. 78-12, 1978

(7) Lee, C.-S., "Prediction of Steady and Unsteady Performance of Marine Propellers with or without Cavitation by Numerical Lifting-Surface Theory", Ph.D. Thesis, MIT, 1979

(8) Huang, T.T., et al., "Propeller/Stern/Boundary-Layer Interaction on Axisymmetric Bodies: Theory and Experiment", DTNSRDC Report No. 76-0113, 1976

(9) Huang, T.T., and Groves, N.C., "Effective Wake: Theory and Experiment", 13th Symposium on Naval Hydrodynamics, Tokyo, Japan, 1980

(10) Bjarne, E., "US Navy Oiler (AO-177 Class) - Model Tests in SSPA Cavitation Tunnel #2 - Design Propeller", Swedish Maritime Research Center, Goteborg, Sweden, DTNSRDC Report No. 2564-1, 1980

Thesis
J2725
c.1

Jamison 199476

Evaluation of a highly skewed propeller for a Naval auxiliary (AO-177).

Thesis
J2725
c.1

Jamison 199476

Evaluation of a highly skewed propeller for a Naval auxiliary (AO-177).

thesJ2725
Evaluation of a highly skewed propeller
3 2768 002 10014 1
DUDLEY KNOX LIBRARY